Prognoza kiše

Seminarski rad u okviru kursa
Istraživanje podataka
Matematički fakultet

Nikola Grulović

6. juni 2019

Sažetak

U svetu postoji sve veće interesovanje za istraživnaje, obradu, rukovanje podacima u različite svrhe, stoga je ovo primer rada u kome je vršeno istraživanje vremenskih podataka u Australiji. Merenjem su prikupljeni podaci o količinama padavina, temperaturi, brzini vetra, vlažnosti vazduha, itd. Pretprocesiranjem, vizuelizacijom i primenom različitih metoda klasifikacije dobijeni su raznoliki rezultati.

Sadržaj

1 Uvod

Merenja o kiši su trajala od 2008 god. do 2017 god. u 49 lokacija u Australiji. Za svaku lokaciju su zabeležene informacije o količini padavina, oblačnosti, sunčanosti, itd. Uz pomoć IBM SPSS Modeler alata, kao i python jezika, u radu će biti predstavljeni različiti rezultati o podacima, koji su dobijeni primenom odgovarajućih algoritama klasifikacije.

2 Podaci

Podaci se mogu preuzeti sa linka. Datoteka sadrži 140 hiljada redova sa 23 atributa, od toga su 17 atributa integer vrednosti, a 6 atributa kategoričke vrednosti. Atributi integer tipa predstavljaju merenja i zapažanja tokom jednog dana. Vrednosti intigera variraju od atributa do atributa. Kod kategoričkih atributa javljaju se nazivi lokacije, datumi merenja i direkcije vetra. Deo podataka se može videti na slici 1.

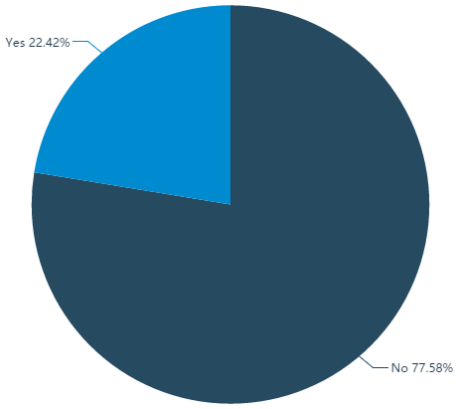
Location	MinTemp	MaxTe...	Rainfall	Evapor...	Sunshine	WindG...	WindG...	WindDi...	WindDi...	WindS...
Albury	10.5	28.4	0	NA	NA	SE	33	SE	SW	19
Albury	11.3	32.2	0	NA	NA	WNW	28	ENE	SSW	17
Albury	13.9	36.6	0	NA	NA	WNW	39	SSE	NNE	2
Albury	18.6	39.9	0	NA	NA	NNW	61	SSE	WNW	9
Albury	19.3	38.1	0.8	NA	NA	NNW	61	NE	WSW	15
Albury	24.4	34	0.6	NA	NA	NW	98	N	NNW	26
Albury	18.8	35.2	6.4	NA	NA	WNW	52	S	NW	6

Slika 1: Podaci

2.1 Opšti podaci o prognozi

Uz pomoć dijagrama će biti predstavljeni neki statistički podaci atributa.

- Predviđanja za kisu tokom merenja su iznosila 22.42% za 'Da' i 77.38% za 'Ne' (Slika 2).
- Atributi 'Evaporation', 'Sunshine', 'Cloud9am' i 'Cloud3pm' imaju 43%, 48%, 38%, 40% nedostajućih vrednosti



Slika 2: Atribut RainTomorrow

2.2 Nedostajuće vrednosti

U samim podacima je bilo nedostajućih vrednosti (Tabela 1). Iz prikaza koliko nedostajućih vrednosti ima može se primetiti da je najviše takvih vrednosti bilo u vezi sa atributima gde je trebalo da se koriste speicjalni instrumenti (Evaporation) ili da se vrši merenje bez instrumenata(Cloud9am, Cloud3pm, Sunshine).

S obzirom da navedena 4 atributa imaju više od 35% nedostajućih vrednosti, oni neće učestvovati u daljem istraživanju. Za preostale atribute bilo je jako malo nedostajućih vrednosti, one su obrađene različitim metodama.U programskom jeziku pzthon kategoričke vrednosti su postavljene na vrednosti koje se najviše puta pojavljuju u tom atributu, dok za integer vrednosti je pronađena njihova srednja vrednost i postavljena na nju.

Tabela 1: Nedostajuće vrednosti

Atribut	**Procenat nedostajućih vrednosti**
Date	0%
Location	0%
MinTemp	0%
MaxTemp	0%
Rainfall	1%
Evaporation	43%
Sunshine	48%
WindGustDir	7%
WindGustSpeed	7%
WindDir9am	7%
WindDir3pm	3%
WindSpeed9am	1%
WindSpeed3pm	2%
Humidity9am	1%
Humidity3pm	3%
Pressure9am	10%
Pressure3pm	10%
Cloud9am	38%
Cloud3pm	40%
Temp9am	1%
Temp3pm	2%
RainToday	1%
RainTomorrow	0%

2.3 Korelacija među atributima

U podacima između atributa postoji veza. Vezu pronalazimo pomoću Pirsonovog koeficijnta korelacije i u nastavku teksta obrađujemo atribute sa koeficijentom većim od 0.8. Kod Drveta odlučivanja parovi atributa u korelaciji nisu izostavljani, dok kod ostalih algoritama, radi njihovog ubrzanja jedan od parova je isključen.

Koeficijent korelacije je računat pomoću IBM SPSS Modeler alata i

programskog jezika python. Sledeći kod prikazuje pronalaženje i izbacivanje jednog od atributa sa visokim koeficijentom korelacije.

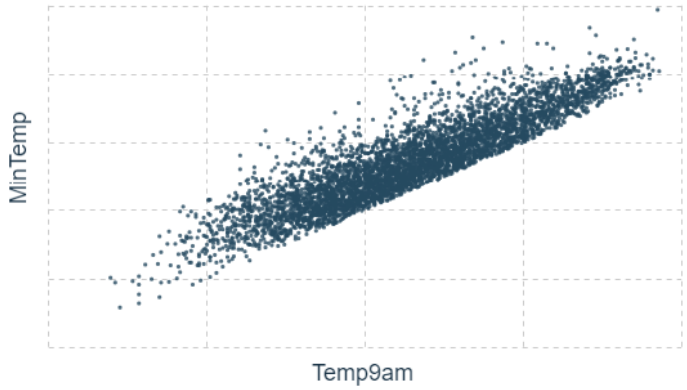
```
corel = df.corr( method='pearson' )
features = corel.columns[:-1].tolist()
elements_in_corelation = {}
for col1 in features:
    for col2 in features:
        if col1 != col2 and abs(corel.loc[col1,col2]) >= 0.8 and
     not elements_in_corelation.get(col2):
            elements_in_corelation[col1] = col2
df = df.drop(columns = list(elements_in_corelation.keys()))
```

Listing 1: Kod za traženje korelacije između atributa

2.3.1 Korelacija između 'MinTemp' i 'Temp9am'

Koeficijent korelacije između atributa 'MinTemp' i 'Temp9am' iznsio je 0.901. Vizuelno, korelacija je prikazana na slici 3.

Posmatranjem ova dva atributa, dolazi se do zapažanja da je najniža temperatura u ranim jutarnjim časovima.

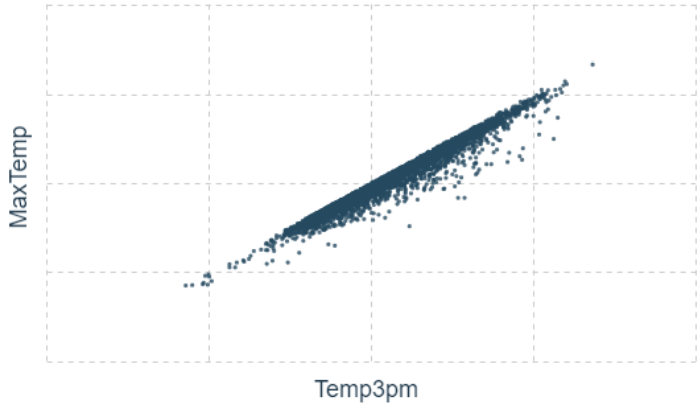


Slika 3: 'MinTemp' i 'Temp9am'

2.3.2 Korelacija između 'MaxTemp' i 'Temp3pm'

Koeficijent korelacija između atributa 'MaxTemp' i 'Temp3pm' iznsio je 0.979. Vizuelno, korelacija je prikazana na slici 4.

Kako je najniža temperatura u ranim jutarnjim satima, iz ove korelacije zapažamo da je dan najtopliji u 15:00 časova.

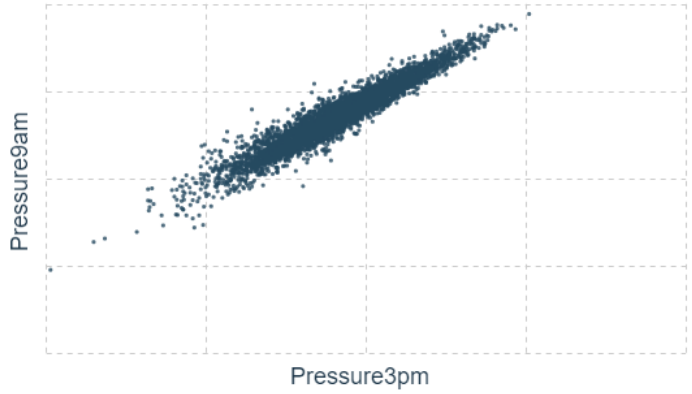


Slika 4: 'MaxTemp' i 'Temp3pm'

2.3.3 Korelacija između 'Pressure9am' i 'Pressure3pm'

Koeficijent korelacija između atributa 'Pressure9am' i 'Pressure3pm' iznsio je 0.957. Vizuelno, korelacija je prikazana na slici 5

. Iz ove korelacije zaključujemo da se pritisak vazduha znatno ne menja izmedjue 9:00 i 15:00.



Slika 5: 'Pressure9am' i 'Pressure3pm'

2.3.4 Korelacija između 'Rainfall' i 'RainToday'

Iz zapažanja i iz samog opisa podataka, atribut RainToday zaivisi od Rainfall. Ako je atribut Rainfall $>$ 1.0, RainToday se postavlja na 1 ('Da'), a u drugom slučaju na 0('Ne').

2.4 Predpocesiranje podataka u Python-u

Kako algoritmi iz biblioteke scikit-learn[3] nemaju automatso predprocesiranje podataka kao IBM SPSS Modeler, predprocesiranje se radi manuelno. Pomoću sledećih kodova se vrši predprocesiranje podataka.

Učitavanje skupa smo vršili pomoću Pandas [2] biblioteke, i čuvali u DataFrame strukturi. Svako pojavljivanje stringa ŇAÏ praznog stringa menjamo sa NaN vrednošću iz Nupmhy[1] biblioteke np.nan. U slučaju da su se pojavili duplikati u podacima, oni bivaju izbrisani.

```
df = pd.read_csv('../Data/weatherAUS.csv'))
features = df.columns[1:-1].tolist()
df.drop_duplicates(inplace = True)
df.replace("NA", np.nan, inplace = True)
df.replace("", np.nan, inplace = True)
```

Listing 2: Učitavanje skupa

Nedostajuće vrednosti za integer smo menjali sa srednjom vrednošću atributa iz određene kolone, dok smo stringove zamenjivali sa najčešćim stringovima iz date kolone.

```
"""
Replacing NaN values
integer type - change to column mean
string type - change to most occuring string
"""
for col in features:
    if df[col].isna().sum() != 0:
        if isinstance(df[col].mode()[0], (np.float64)):
            df[col].replace(np.nan, df[col].mean(), inplace = True
    )
        else:
            df[col].replace(np.nan, df[col].mode()[0], inplace =
    True)
```

Listing 3: Uklanjanje NaN vrednosti

Elemente van granica smo eliminisali iz skupa formula za kvantile, tj ako se element nalazio van $q_1 - 1.5(q_3 - q_1)$ donje granice ili van $q_3 + 1.5(q_3 - q_1)$ gornje granice.

```
"""
Removing elements outside the boundaries
"""
for col in features:
    val = df[col].head(1).values[0]
    if isinstance(val, (np.float64)):
        q1 = df[col].quantile(0.25)
#         q2 = df[col].quantile(0.5)
        q3 = df[col].quantile(0.75)
        ext = [q1-1.5*(q3-q1),q3+1.5*(q3-q1)]
        df.drop( df[(df[col]<ext[0]) | (df[col]>ext[1])].index,
    inplace = True)
```

Listing 4: Skidanje elementa van granica

Kako algoritmi bibliotke sckit-learn rade samo sa numeričkim podacima, vršimo tranformaciju kategoričkih u numeričke preslikavanjem svakog kategoričkog u određen ceo broj.

```
"""
Conversion of String elements into numeric
"""
string_elemets = ['Location','WindGustDir','RainTomorrow']
for element in string_elemets:
    list_of_elements = list(set(df[element]))
    df.replace(list_of_elements, list(range(0,len(list_of_elements
    ))), inplace = True)
```

Listing 5: Transformacija u numeričke

Vršimo min-max normalizaciju zbog atributa Rainfall koji ima opseg od [0.0 - 250.0].

```
"""
Nomrlazing data
"""
x = pd.DataFrame(prep.MinMaxScaler().fit_transform(df[features]))
x.columns = features
```

Listing 6: Normalizacija

3 Klasifikacija

Klasifikacija predstavlja pronalaženje modela koji preslikava skup X u ciljni skup Y ['Da','Ne']. U skupu podataka se javljaju 6 kategoričkih atributa. Predprocesiranjem brišemo elemente van granica iz skupa u python-u, dok u IBM SPSS Modeleru koristimo ugrađene metode.

3.1 Drveta odlučivanja

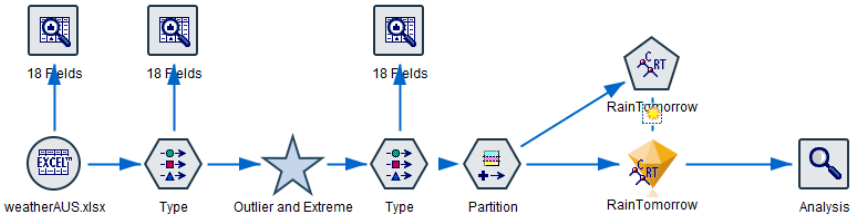
Rezultati koji su se dobijali primenom različitih algoritama su skoro pa ekvivalentni. Algoritam C&RT i python su davali nešto lošije rezultate u odnosu na algoritam C5.0. U IBM SPSS Modeleru i u pythonu smo koristili unakrsnu validaciju. Skupovi su bili podeljeni na trening(70%) i test(30% u pythonu i 20% u SPSS Modeleru) skupove.

3.1.1 C&RT

U IBM SPSS Modeler-u tok podatakaje izgledao kao na slici 6. Nakon učitavanja podataka, pronađeni su elementi van granica i ekstremi, obrađeni su tako što su elementi van granica zamenjeni srednjom vrednošću, a ekstremi pretvoerni u NaN. Nakon toga model je napravljen algoritmom C&RT.

Parametri koji su bili podešeni prilikom pravljenja modela su sledeći:

- Maximum Tree Depth : 8
- Missclassification cost : Actual Yes 2.0
- Impurity messure : Gini



Slika 6: IBM SPSS C&RT

Nakon primene skupa podataka na model, dobijaju se sledeći rezultati dati u tabelama 2, 3.

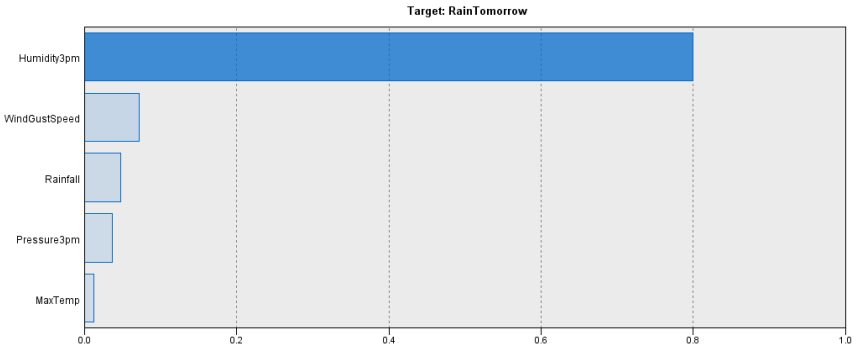
Tabela 2: Rezultati modela

Partition	Training	Testing	Validation
Correct	81,399 | 81.64%	23,089 | 81.69%	11,658 | 81.93%
Wrong	18,302 | 18.36%	5,174 | 18.31%	2,571 | 18.07%
Total	99,701	28,263	14,229

Tabela 3: Matrica konfuzije

	No	Yes
No	68,994	7,941
Yes	9,805	12,388

Preciznost modela iznosi oko 82.6%. Dubina stabla je iznosila 5. Iz rezultata kao i iz podatak se zaključujemo da je instanca 'Ne' ciljnog atributa dominantnija u odnosu na 'Da'. Iz modela zapažamo da je atribut Humidity3pm jedan od najvažnijih atributa i najviše utiče na određivanje ciljne klase(Slika 7)



Slika 7: Važnost atributa

3.1.2 C5.0

Tok podataka u IBM SPSS Modeleru je skoro pa identičan kao kod C&RT obrade. Suštinska razlika je obrada NaN vrednosti za integer tipove i belina za string tipove. Stringovi su obrađenni tako što je nedostajuća vrednost zamenjena sa vrednošću koja se najčešće pojavljuje u podacima za taj atribut sa izuzetkom RainToday gde smo na random način dodeljivali vrednosti. Integer vrednosti su zamenjene uz pomoć C&RT algoritma.

Parametri koji su bili podešeni prilikom pravljenja modela su sledeći:

- Cross-Validate : 5
- Missclassification cost : Actual Yes 2.0
- Favor : Accuracy

Nakon primene skupa podataka na model, dobijaju se sledeći rezultati dati u tabelama 4, 5.

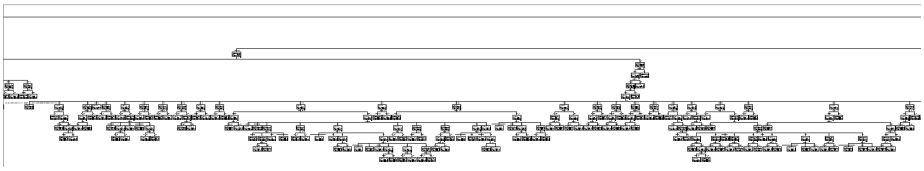
Tabela 4: Rezultati modela

Partition	Training	Testing	Validation
Correct	85,636 | 86.04%	24,521 | 82.26%	12,221 | 85.86%
Wrong	13,893 | 13.96%	3,907 | 13.74%	2,021 | 14.14%
Total	99,532	28,428	14,233

Tabela 5: Matrica konfuzije nad trening skupom

	No	Yes
No	69,675	7,443
Yes	6,450	15,964

Preciznost modela iznosi oko 83%. Dubina stabla je iznosila 8. C5.0 je precizniji u odnosu na C&RT zbog poće dubine stable, pa samim tim stablo je komplikovanije (Slika 8). Kao i kod C&RT algoritma atribut sa najvećom važnošću je bio Humidity3pm.



Slika 8: Deo stabla C5.0 algoritma

3.1.3 Python

Nakon pretprocesiranja podataka i podele na trening i test skupove, primenom algoritma (Kod 3.1.3) dobijaju se sledeći rezultati. U tabeli 6 se nalaze rezultati modela nad trening skupom, a odgovarajuća matrica konfuzije je data u tabeli 7. Nad test skupom rezultati su dati u tabeli 8 sa odgovarajućom matricom konfuzije u tabeli 9.

Drveta odlučivanja koja daju približno iste rezultate imaju dubinu stabla između 8 i 10. Iz rezultata zaključujemo da se bolje klasifikuje vrednost atribute 'Ne', što je i očekivano zbog disbalansa atributa RainTomorrow.

```
x_train, x_test, y_train, y_test = train_test_split(x, y,
    train_size = 0.7)
parameters = {
        'max_depth' : range(4,12)
        }

tree = GridSearchCV(tree.DecisionTreeClassifier(),
                    parameters,
                    cv = 5,
                    scoring = 'f1_macro')

tree.fit(x_train, y_train)

y_pred = tree.predict(x_train)
print('Precision', met.accuracy_score(y_train, y_pred))
print(met.classification_report(y_train, y_pred))
cnf_matrix = met.confusion_matrix(y_train, y_pred)
print("Confusion Matrix", cnf_matrix, sep="\n")

y_pred = tree.predict(x_test)
print('Precision', met.accuracy_score(y_test, y_pred))
print(met.classification_report(y_test, y_pred))
cnf_matrix = met.confusion_matrix(y_test, y_pred)
print("Confusion Matrix", cnf_matrix, sep="\n")
```

Listing 7: Drvo odlučivanja

- Trening Skup
  Preciznost nad celim trening skupom je iznsila oko 89.56% (0.8956)

Tabela 6: Rezultati nad trening skupom

Class	**precision**	**recall**	**f1-score**	**support**
Yes	0.61	0.34	0.43	5004
No	0.90	0.98	0.94	63633

Tabela 7: Matrica konfuzije

	Yes	**No**
Yes	4552	6804
No	1223	61609

- Test Skup
  Preciznost nad celim test skupom je iznsila oko 86.29% (0.8629)

Tabela 8: Rezultati nad test skupom

Class	**precision**	**recall**	**f1-score**	**support**
No	0.89	0.96	0.92	27237
Yes	0.61	0.34	0.43	5004

Tabela 9: Matrica konfuzije

	Yes	**No**
Yes	1498	3369
No	3942	25986

3.2 Metod potpornih vektora

Pravljenje modela metodom potpornih vektora je dosta procesorski zahtevnije od drveta odlučivanja pa će mo vaditi uzorak skupa i praviti model nad njim. Izdvojen je uzorak veličine 30%, takođe je smanjena dimenzionalnost pomoću prethodno izračunatih korelacija. Pored korelacija je korišćen metod PCA za smanjenje dimenzionalnosti.

Nedostajuće vrednosti su zamenjene odgovarajućim pomoću C&RT algoritma, dok su ekstremi odbačeni a elementi van granica zamenjeni srednjom vrednošću odgovarajućih atributa. Za prvaljenje modela je izabran polinomjalni kernel sa parametrom C koji iznosi 8.

Slika 9: IBM SPSS SVM

Rezultati koji se dobijaju su u tabeli 10 sa odgovarajućom matricom konfuzije nad trening skupom 11 i nad test skupom 12. Preciznost modela je iznosila 83.37% nad smanjenom skupu, a nad originalnom 70.42%. Iz rezultata zaključujemo da se instanca 'Ne' atributa RainTomorrow bojle klasifikuje od instance 'Da' što oslikava manjinu instanci 'Da' u orginalnom skupu.

Tabela 10: Rezultati modela

Partition	**Training**	**Testing**
Correct	20,943 | 83.32 %	9,070 | 83.42 %
Wrong	4,193 | 16.68 %	1,803 | 16.58 %
Total	25,163	10,873

Tabela 11: Matrica konfuzije nad tretning skupom

	No	**Yes**
No	17,268	2,310
Yes	1,883	3,675

Tabela 12: Matrica konfuzije nad test skupom

	No	**Yes**
No	7,458	991
Yes	812	1,612

Prilikom korišćenja metoda PCA za smanjenje dimenzionalnosti, dobija se ubrazanje od 330%. Izabranih 6 atributa objašnjavaju 96.5% skupa. (Slika 10).

Rezultati modela se nalaze u tabeli 13. Postignuta preciznost iznosi 84.21%. Zaključujemo da primenom PCA tehnike smanjenja dimenzionalnostii znatno se ubrzavamo pronalaženje ciljnog atributa, ali sa cenom da gubimo neke informacije u podacima.

Total Variance Explained

Component	Initial Eigenvalues			Extraction Sums of Squared Loadings		
	Total	% of Variance	Cumulative %	Total	% of Variance	Cumulative %
1	3.142	39.277	39.277	3.142	39.277	39.277
2	2.053	25.653	64.935	2.053	25.658	64.935
3	1.140	14.247	79.181	1.140	14.247	79.181
4	.586	7.329	86.510	.586	7.329	86.510
5	.489	6.115	92.625	.489	6.115	92.625
6	.310	3.874	96.499	.310	3.874	96.499
7	.246	3.075	99.575			
8	.034	.425	100.000			

Extraction Method: Principal Component Analysis.

Slika 10: PCA

Tabela 13: Rezultati modela

Partition	**Training**	**Testing**
Correct	21,651 | 84.14 %	9,090 | 84.28 %
Wrong	4,081 | 15.86 %	1,696 | 15.72 %
Total	25,163	10,873

U programskom jeziku python, tražili smo najbolje parametre za model pomoću uzoračkog skupa.. Uzorački skup je sadržao 20% originalnog skupa,

```
parameters = [{'C' : [1,5],
               'kernel' : ['poly'],
               'degree' : [1,3],
               'gamma' : [0.5,1],
               'coef0' : [.5]
            },
            {
               'C' : [10],
               'kernel' : ['rbf'],
               'coef0' : [.5,1]
            }]
clv = GridSearchCV(svm.SVC(),parameters, cv=3, scoring='f1_macro')
clv.fit(x_train, y_train)
```

Listing 8: SVM parametri

Rezultati koje dobijamo nakon izvršavanja koda su:

- C : 5
- coef0 : 0.5
- degree : 3
- gamma : 1
- kernel : poly

Preciznost nad trening skupom je iznosila 87.7% sa propratnim rezultatima datim u tabeli 14, a nad test skupom preciznost je iznosila 87.2% sa rezultatima datim u tabeli 15. Matrica konfuzije nad trening skupom data je u tabeli 16 i nad test skupom u tabeli 17.

Iz rezultata možemo zaključiti da je uzorak sadržao veći broj instanci 'Ne', i samim tim prave instance 'Da' lošije klasifikovano, tačnije klasifikovane su kao 'Ne'.

Tabela 14: Rezultati nad trening skupom

Class	**precision**	**recall**	**f1-score**	**support**
Yes	0.80	0.26	0.39	2225
No	0.88	0.99	0.93	12465

Tabela 15: Rezultati nad test skupom

Class	**precision**	**recall**	**f1-score**	**support**
Yes	0.77	0.23	0.35	954
No	0.88	0.99	0.93	5342

Tabela 16: Matrica konfuzije trening skupa

	Yes	**No**
Yes	568	1657
No	140	12325

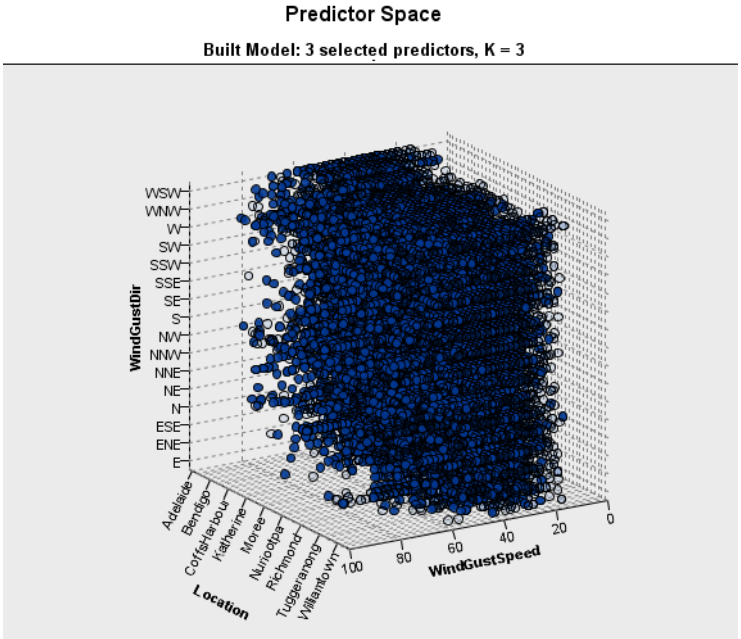
Tabela 17: Matrica konfuzije test skupa

	Yes	**No**
Yes	217	737
No	66	5276

3.3 K-najbližih suseda

Kod algoritma K najbližih suseda kao kod metoda potpornih vektora, smanjujemo dimenzionalnost skupa prethodno izračunatom korelacijom, pravimo uzorački skup koji sadrži 30% originalnog skupa, ekstremne vrednosti brišemo i elemente van granica postavljamo na srednje vrednosti. Nedostajuće vrednosti zamenjujemo vrednostima izračunatim pomoć C&RT algoritma.

Prilikom pravljenja modela postavljamo broj suseda na 3 (K=3), a za distancu euklidsko rastojanje. Model je prikazan na slici 11.



Slika 11: KNN Model

Kao rezultat izvršavanja modela nad podacima dobijaju se sledeći rezultati prikazani u narednim tabelama. Rezultat modela je dat u tabeli 18 i matrice konfuzije nad trening skupom 19 i nad test skupom 20. Ukupna preciznost modela za uzorak je iznosila 81.94%

Tabela 18: Rezultati modela

Partition	**Training**	**Testing**
Correct	20,722 | 82.03 %	8,840 | 81.85 %
Wrong	4,539 | 17.97 %	1,960 | 18.15 %
Total	25,261	10,800

Tabela 19: Matrica konfuzije tretning skupa

	No	**Yes**
No	18,367	1,339
Yes	3,200	2,355

Tabela 20: Matrica konfuzije test skupa

	No	**Yes**
No	7,836	531
Yes	1,429	1,004

Iz matrica kofnuzije, kao i u prethodnim algoritmima, 'Da' kao ciljni atribut se lošije klasifikuje od 'Ne', zbog neizbalansiranosti ciljnog atributa.

U programskom jeziku python, tražili smo najbolje parametre nad našim skupom prilikom pravljenja modela. Skup je bio uzorak od 30% originalnog skupa. Kao rezultat izvršavanja koda 3.3, dobijaju se sledeći paramteri:

- n_neighbors = 4
- p = 2
- weights = 'uniform'

```
parameters = {
        'n_neighbors' : range(3,6),
        'p' : [1,2],
        'weights' : ['distance','uniform']
        }

knn = GridSearchCV(KNeighborsClassifier(), parameters, cv = 5,
    scoring = 'f1_macro')
knn.fit(x_train, y_train)
```

Listing 9: KNN parametri

Rezultati koji se dobijaju su dati u sledećim tabelama. Iz samih rezultata možemo da primetimo da je došlo do malog predprilagođavanja podataka, tj. nad trening skupom preciznost modela je iznosila 90% dok je nad test skupom iznosila 83%.

Možemo da primetimo da se 'Da' malo lošije klasifikuje od 'Ne' u trening skupu, dok se u test skupu preciznost 'Da' drastično smanjila, tačnije iznosi 47%. To se može i videti iz matrica konfuzija.

Tabela 21: Rezultati nad trening skupom

Class	**precision**	**recall**	**f1-score**	**support**
Yes	0.70	0.60	0.65	3408
No	0.93	0.95	0.94	18665

Tabela 22: Rezultati nad test skupom

Class	**precision**	**recall**	**f1-score**	**support**
Yes	0.47	0.37	0.41	1461
No	0.89	0.92	0.91	8000

Tabela 23: Matrica konfuzije trening skupa

	Yes	**No**
Yes	2048	1360
No	888	17777

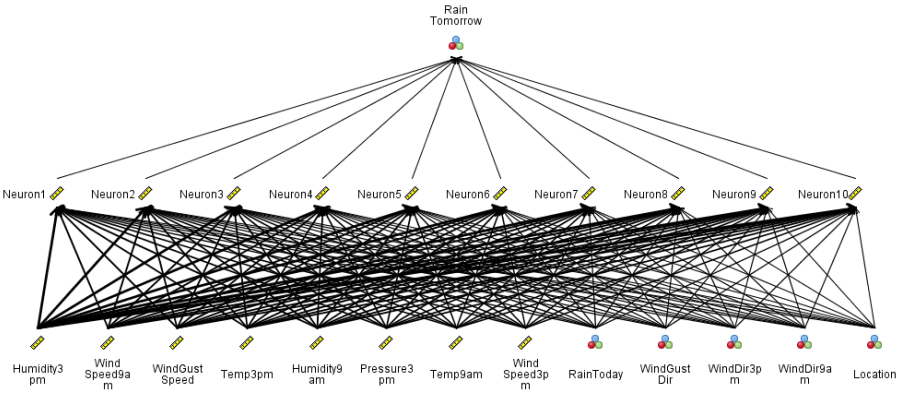
Tabela 24: Matrica konfuzije test skupa

	Yes	**No**
Yes	539	922
No	618	7382

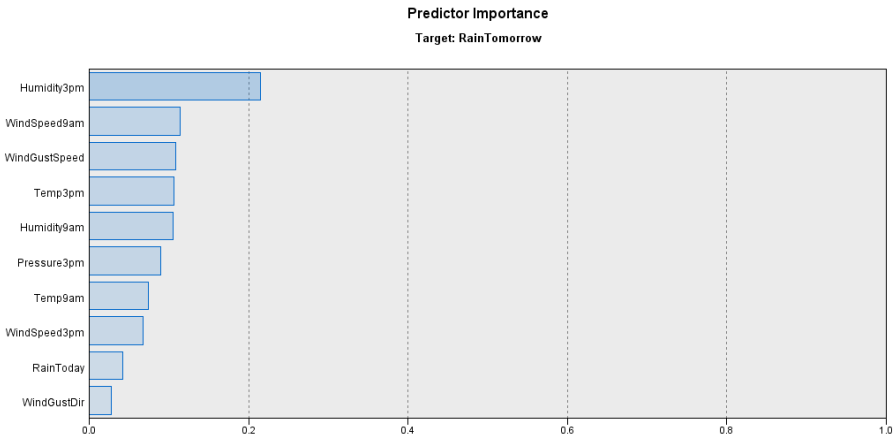
3.4 Veštačke neuronkse mreže

Za pravljenje modela neuronskih mreža, dimenzionalnost podataka smanjujemo prethodno izračunatim korelacijama, nedostajuće vrednosti računamo i dopunjujemo pomoću C&RT algoritma, elemente van granica zamenjujemo odgovarajućim srednjim vrednostima i ekstremne vrednosti nuliramo.

Kao rezultat dobijamo model prikazan na slici 12 i odgovarajuće vrednosti date u tabeli 25. Neuronske mreže imaju 1 skriveni sloj. Koristimo RBF model za neuronske mreže. Najvažniji atribut je Humidity3pm (Slika 13).



Slika 12: Neuronske mreže



Slika 13: Važnost atributa

Tabela 25: Rezultati modela

Partition	**Training**	**Testing**
Correct	69,447 | 82.5 %	30,097 | 82.64 %
Wrong	14,734 | 17.5 %	6,322 | 17.36 %
Total	84,181	36,419

Tabela 26: Matrica konfuzije nad tretning skupom

	No	**Yes**
No	62,751	2,734
Yes	12,000	6,696

Tabela 27: Matrica konfuzije nad test skupom

	No	**Yes**
No	27,231	1,206
Yes	5,116	2,866

Preciznost modela je iznosila 82.57 %, sa tim da se instanca 'Da' atributa RainTomorrow skoro duplo više klasifikuje kao 'Ne', to se može videti i iz matrica konfuzije (Tabele 26 i 27).

U radu sa mrežama u pythonu, postavljamo sledeće paramtre i trazimo najbolji paramtar. Za testiranje ovih parametara uzimamo uzorak od 20% iz originalnog skupa. Kao rezultat se dobija:

- solver : adam
- activation : relu
- learning_rate : adaptive

```
hidden_layer_sizes = []
for i in range(1,2):
    for j in range(8,12):
        hidden_layer_sizes.append((j,)*i)
params = [{'solver': ['sgd','adam'],
          'learning_rate': [ 'adaptive','constant'],
          'activation': [ 'relu', 'logistic'],
          'hidden_layer_sizes': hidden_layer_sizes,
          'max_iter' : [1000]
          }]
clf = GridSearchCV(MLPClassifier(), params, cv=5)
clf.fit(x_train, y_train)
```

Listing 10: NN parametri

Nakon pronalaženja najboljih paramtetara, treniramo model nad celim skupom. Rezultat je prikazan u sledećoj tabeli za trening skup 28 sa matricom konfuzije 30 i za test skup 29 sa matricom konfuzije 31.

Tabela 28: Rezultati nad trening skupom

Class	**precision**	**recall**	**f1-score**	**support**
Yes	0.70	0.29	0.41	11356
No	0.88	0.98	0.93	62832

Tabela 29: Rezultati nad test skupom

Class	**precision**	**recall**	**f1-score**	**support**
Yes	0.70	0.28	0.40	4867
No	0.88	0.98	0.93	26928

Tabela 30: Matrica konfuzije trening skupa

	Yes	**No**
Yes	3251	8105
No	1399	61433

Tabela 31: Matrica konfuzije test skupa

	Yes	**No**
Yes	1385	3482
No	592	26336

4 Zaključak

Primenom algoritama klasifikacije dobija se približno ista preciznost modela različitih algoritama. Međutim neki algoritmi daju bolje rezultate od drugih, neki bolje klasifiku instancu 'Da', a neki su dosta brži prilikom pravljenja modela. Algoritam koji je najbolje klasifikovao instacu 'Da' je bio C5.0 sa preciznošću od 83%, dok algoritam za drveta odlučivanja u pythonu je imao najveću preciznost od 87%.

Ono što možemo da zaključimo svaki algoritam je davao prilično dobre rezultate sa malim odstupanjima, bilo koji metod da se izabere, na kraju će model dobro klasifikovati ciljnu klasu, tačnije predviđanje o sutrašnjoj prognozi.

Literatura

[1] Numpy. Online at: https://www.numpy.org/.

[2] Pandas. Online at: https://pandas.pydata.org/.

[3] Scikit learn. Online at: https://scikit-learn.org/.