Univerzitet Singidunum
Tehnički fakultet

Predviđanje dovoljnosti mesečne proizvodnje solarnog sistema u R programskom jeziku

- Projektni rad -

Predmet: **Praktikum – Napredno softversko inženjerstvo**

Profesor:
prof. dr Angelina Njeguš

Asistent:
Petar Biševac, master

Student:
Marko Dojkić 2018/201682

Beograd, 2022. godine

Sadržaj

1. Analiza skupa podataka 3
2. Čišćenje skupa podataka 4
3. Smanjenje dimenzionalnosti i vizualizacija podataka 6
4. Primena algoritma mašinskog učenja – Random forest 10
5. Zaključak 12

1. Analiza skupa podataka

Skup podataka predstavlja podatke o izlaznoj snazi horizontalnih fotonaponskih panela koji se nalaze na 12 lokacija na severnoj hemisferi tokom 14 meseci. Ovaj skup podataka je korišćen za potrebe predviđanja izlazne snage panela i predstavljen u radu "Machine Learning Modeling of Horizontal Photovoltaics Using Weather and Location Data".

Za potrebe ovog projektnog rada biće pokušano predviđanje da li je mesečna proizvodnja solarnog sistema na osnovu dostavljenih parametara o vremenu biti dovoljna za domaćinstvo na datim lokacijama. Kako nam nisu potrebni svi atributi koji su nam na raspolaganju u nastavku ću opisati samo one koji su uneseni u algoritam mašinskog učenja (postupak čišćenja podataka će biti prikazan u nastavku projektnog rada):

- location – Naziv mesta gde se nalazi solarna elektrana*
- month – Mesec u godini kada je izmerena proizvodnja*
- humidity – Posečna** vlažnost vazduha u procentima
- temp – Posečna** temperatura (°C)
- wind_speed – Posečna** brzina vetra (km/h)
- visibility – Posečna** vidljivost (km)
- pressure – Prosečan** atmosferski pritisak (milibar)
- power_output – Posečna** proizvodnja solarnog sistema (wh)
- cloud_coverage – Posečna** pokrivenost oblacima u procentima
- is_enough_power_produced – Logička vrednost koju razmatramo (uslov: power_output > 1000.0)

** Podaci su grupisani na osnovu ova dva atributa.*
*** Prilikom obrade podataka za ove atribute uzeta je njihova prosečna mesečna vrednost za svaku lokaciju.*

U nastavku je prikazana struktura skupa podataka i izgled prvih 10 redova.

```
> str(df)
'data.frame':   128 obs. of  10 variables:
 $ location               : chr  "Camp Murray" "Camp Murray" "Camp Murray" "Camp Murray" ...
 $ month                  : int  12 1 2 3 4 5 6 7 8 9 ...
 $ humidity               : num  42.49 59.86 58.44 34.31 5.92 ...
 $ temp                   : num  21.8 18 15.7 24.4 47.8 ...
 $ wind_speed             : num  2.55 8.02 7.82 7.36 9.67 ...
 $ visibility             : num  7.02 8.79 9.05 10 10 ...
 $ pressure               : num  1019 1007 1009 999 1016 ...
 $ power_output           : num  184.4 301.4 916.3 207.2 59.2 ...
 $ cloud_coverage         : num  484 267 229 351 722 ...
 $ is_enough_power_produced: logi  FALSE FALSE FALSE FALSE FALSE TRUE ...
```

Slika 1: Prikaz structure skupa podataka

	location	month	humidity	temp	wind_speed	visibility	pressure	power_output	cloud_coverage	is_enough_power_produced
1	Camp Murray	12	42.48726	21.83310	2.547170	7.020755	1019.196	184.40258	484.2453	FALSE
54	Camp Murray	1	59.85867	18.01267	8.023256	8.790698	1007.387	301.42147	266.9186	FALSE
140	Camp Murray	2	58.43724	15.71550	7.824242	9.053939	1008.691	916.26362	229.4364	FALSE
305	Camp Murray	3	34.31480	24.37622	7.363636	10.000000	998.900	207.20663	351.2727	FALSE
327	Camp Murray	4	5.92041	47.76283	9.666667	10.000000	1015.500	59.16473	722.0000	FALSE
330	Camp Murray	5	30.94171	31.78248	5.103774	10.000000	1006.644	1399.37870	175.9340	TRUE

Slika 2: Prikaz prvih 10 redova skupa podataka

Na osnovu ovih podataka biće izvršeno predviđanje upotrebom algoritma nasumičnih stabala odlučivanja (eng. random forest) koji će klasifikovati naše podatke po tome da li ispunjavaju traženi uslov ili ne.

2. Čišćenje skupa podataka

```
> str(data)
'data.frame':   21045 obs. of  17 variables:
 $ Location     : chr  "Camp Murray" "Camp Murray" "Camp Murray" "Camp Murray" ...
 $ Date         : int  20171203 20171203 20171203 20171204 20171204 20171204 20171205 20171205 20171205 20171205 ...
 $ Time         : int  1145 1315 1330 1230 1415 1430 1115 1200 1300 1400 ...
 $ Latitude     : num  47.1 47.1 47.1 47.1 47.1 ...
 $ Longitude    : num  -123 -123 -123 -123 -123 ...
 $ Altitude     : int  84 84 84 84 84 84 84 84 84 84 ...
 $ YRMODAHRMI   : num  2.02e+11 2.02e+11 2.02e+11 2.02e+11 2.02e+11 ...
 $ Month        : int  12 12 12 12 12 12 12 12 12 12 ...
 $ Hour         : int  11 13 13 12 14 14 11 12 13 14 ...
 $ Season       : chr  "Winter" "Winter" "Winter" "Winter" ...
 $ Humidity     : num  81.7 96.6 93.6 77.2 54.8 ...
 $ AmbientTemp  : num  12.87 9.66 15.45 10.37 16.85 ...
 $ PolyPwr      : num  2.43 2.46 4.47 1.65 6.58 ...
 $ Wind.Speed   : int  5 0 5 5 3 0 0 5 5 6 ...
 $ Visibility   : num  10 10 10 2 3 5 4 7 10 10 ...
 $ Pressure     : num  1011 1011 1012 1024 1024 ...
 $ Cloud.Ceiling: int  722 23 32 6 9 722 722 722 722 722 ...
```

Slika 3: Struktura inicijalnog skupa podataka

Na osnovu inicijalne strukture vidimo da postoje atributi koji su redudantni poput geografske širine i dužine, nadmorske vidsine i brojčane vrednosti datuma i vremena merenja. Pored tih podataka višak su i podaci o godišnjem dobu, datumu i vremenu merenja koji nisu od značaja za naš algoritam, pa će isti biti uklonjeni korišćenjem biblioteke “dplyr”.

```
library(dplyr) #Library needed for data manipulation

data <- data %>% select(-Date)
data <- data %>% select(-Time)
data <- data %>% select(-Hour)
data <- data %>% select(-Latitude)
data <- data %>% select(-Longitude)
data <- data %>% select(-Altitude)
data <- data %>% select(-YRMODAHRMI)
data <- data %>% select(-Season)
```

Slika 4: Kod za uklanjanje redudantnih podataka

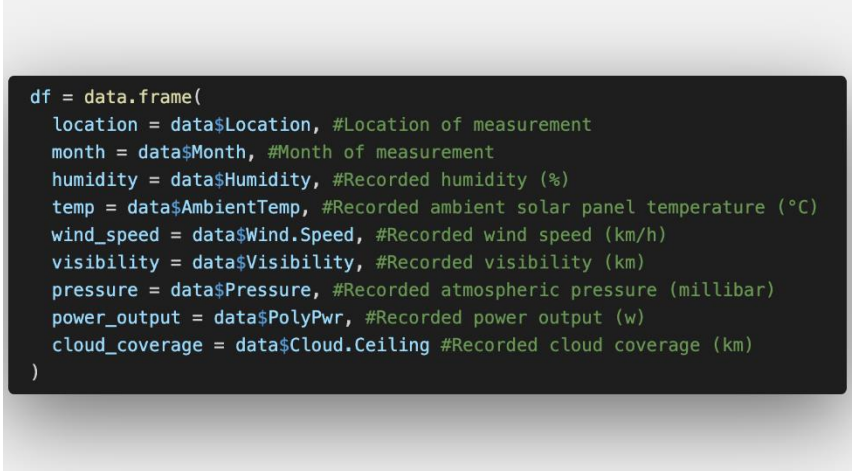
Nakon uklanjanja je potrebno da proveromo da li postoje neispravno uneti podaci.

```
> colSums(is.na(data)) #We don't have missing values in our data, no processing needed
    Location        Month     Humidity  AmbientTemp      PolyPwr   Wind.Speed   Visibility     Pressure Cloud.Ceiling
           0            0            0            0            0            0            0            0            0
```

Slika 5: Provera da li postoje neispravno uneti podaci

Slika 5 pokazuje su svi podaci ispravno uneseni, u suprotnom bilo bi potrebno da dopunimo naš skup podataka.

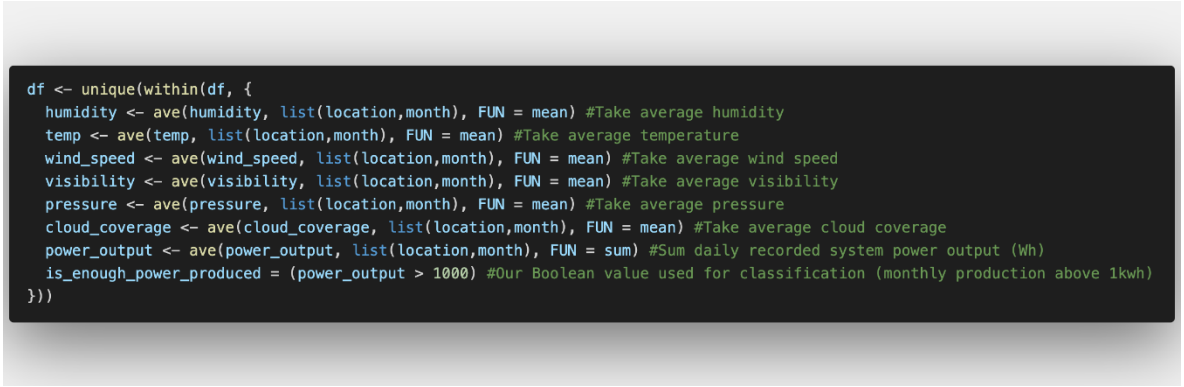
Sada ćemo podatke preimenovati radi lakšeg korišćenja i smestiti u naš „data frame”.



```
df = data.frame(
  location = data$Location, #Location of measurement
  month = data$Month, #Month of measurement
  humidity = data$Humidity, #Recorded humidity (%)
  temp = data$AmbientTemp, #Recorded ambient solar panel temperature (°C)
  wind_speed = data$Wind.Speed, #Recorded wind speed (km/h)
  visibility = data$Visibility, #Recorded visibility (km)
  pressure = data$Pressure, #Recorded atmospheric pressure (millibar)
  power_output = data$PolyPwr, #Recorded power output (w)
  cloud_coverage = data$Cloud.Ceiling #Recorded cloud coverage (km)
)
```

Slika 6: Kod za preimenovanje atributa

Poslednji korak koji je potrebno da primenimo je grupisanje podataka. Kao što je već navedeno u uvodnom delu grupisaćemo podatke po mesecu i logaciji merenja. Izlaznu snagu solarnog sistema ćemo pretvoriti u Wh tako što ćemo sabrati sve vrednosti individualnih merenja u mesecu, dok ćemo za sve ostale brojčane vrednosti uzeti njihovu prosečnu vrednost. U ovom ćemo dodati i logičku vrednost koju razmatramo (**Da li je mesečna proizvodnja veća od 1 kilovat sat?**).



```
df <- unique(within(df, {
  humidity <- ave(humidity, list(location,month), FUN = mean) #Take average humidity
  temp <- ave(temp, list(location,month), FUN = mean) #Take average temperature
  wind_speed <- ave(wind_speed, list(location,month), FUN = mean) #Take average wind speed
  visibility <- ave(visibility, list(location,month), FUN = mean) #Take average visibility
  pressure <- ave(pressure, list(location,month), FUN = mean) #Take average pressure
  cloud_coverage <- ave(cloud_coverage, list(location,month), FUN = mean) #Take average cloud coverage
  power_output <- ave(power_output, list(location,month), FUN = sum) #Sum daily recorded system power output (Wh)
  is_enough_power_produced = (power_output > 1000) #Our Boolean value used for classification (monthly production above 1kwh)
}))
```

Slika 7: Kod za grupisanje podataka

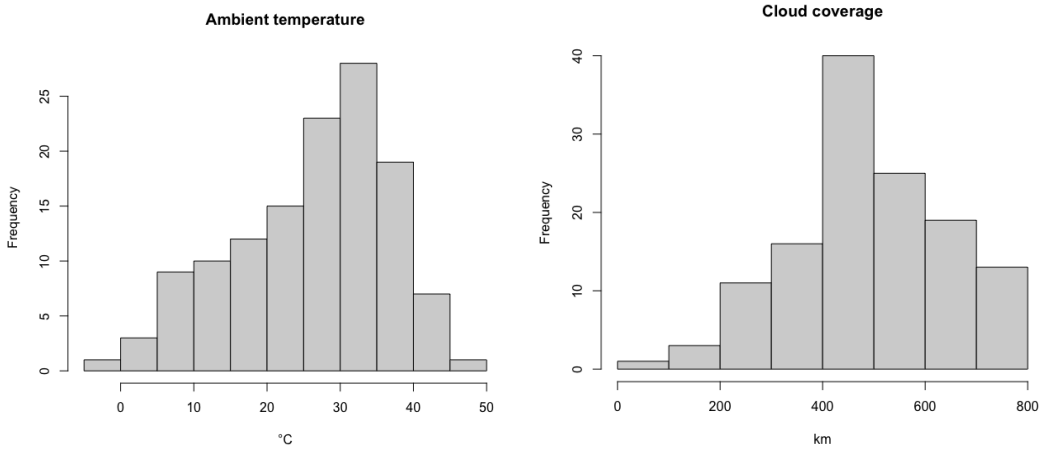
Nakon ovoga dobijamo skup podataka koji prikazan na slikama 1 i 2.

3. Smanjenje dimenzionalnosti i vizualizacija podataka

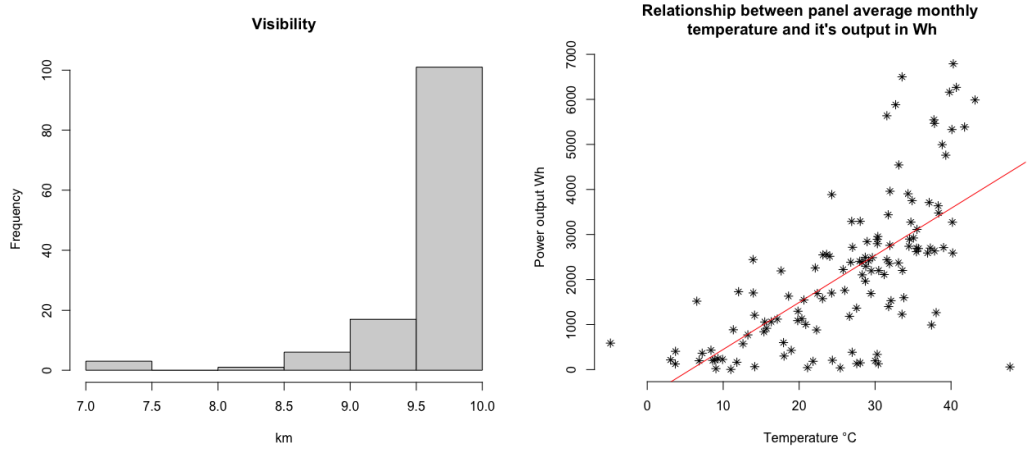
Kada smo sredili podatke sada ćemo videti histograme koji pokazuju prosečnu temperature, pokrivenost oblacima i vidljivost. Takođe ćemo videti diagram odnosa temperature i proizvedene količine električne energije, čiju zavisnost ćemo proveriti Pirsonovim koeficijentom korelacije.

```r
hist(df$temp, main = "Ambient temperature", xlab="°C")
hist(df$cloud_coverage, main = "Cloud coverage", xlab="km")
hist(df$visibility, main = "Visibility", xlab="km")

#In already conducted study mentioned on line 11 of this code it is proven that greatest impact on
power output has temperature and thus
#we will show relationship between power output and average temperature

a <- df$temp
b <- df$power_output

plot(a, b, main = "Relationship between panel average monthly\n temperature and it's output in Wh"
, xlab = "Temperature °C", ylab = "Power output Wh", pch = 8, frame = FALSE)
abline(lm(b ~ a, data = df), col = "red")
```

Slika 8: Kod za prikaz histograma i grafikona odnosa temperature i proizvedene količine električne energije



Slike 9 i 10: Histogrami koji prikazuju zastupljenost prosečne temperature i procentualne pokrivenosti oblacima



Slike 11 i 12: Histogram prosečne vidljivosti i diagram odnosa temperature i proizvedene količine električne energije

Nakon primene koda sa slike 13, prvo vidimo sa slike 14 da dobijamo pozitivnu korelaciju od 0.6623996, što pokazije pozitivnu zavisnost između temperature i proizvedene količine električne energije.

```r
cor.test(a, b, method = c("pearson", "kendall", "spearman"))
#The p-value of the test is 2.2^{-16}, which is less than the significance level alpha = 0.05.
#We can conclude that panel temperature and it's output are significantly correlated
#with a correlation coefficient of 0.66 and p-value of 2.2^{-16}.

summary(df) #Display statistical informations about data
```

Slika 13: Kod za prikaz rezultata Pirsonovog testa korelacije i rezimea statističkih parametara

```
> cor.test(a, b, method = c("pearson", "kendall", "spearman"))

        Pearson's product-moment correlation

data:  a and b
t = 9.9251, df = 126, p-value < 2.2e-16
alternative hypothesis: true correlation is not equal to 0
95 percent confidence interval:
 0.5523611 0.7497491
sample estimates:
      cor
0.6623996
```

Slika 14: Rezultat Pirsonovog testa korelacije između temperature i proizvedene količine električne energije

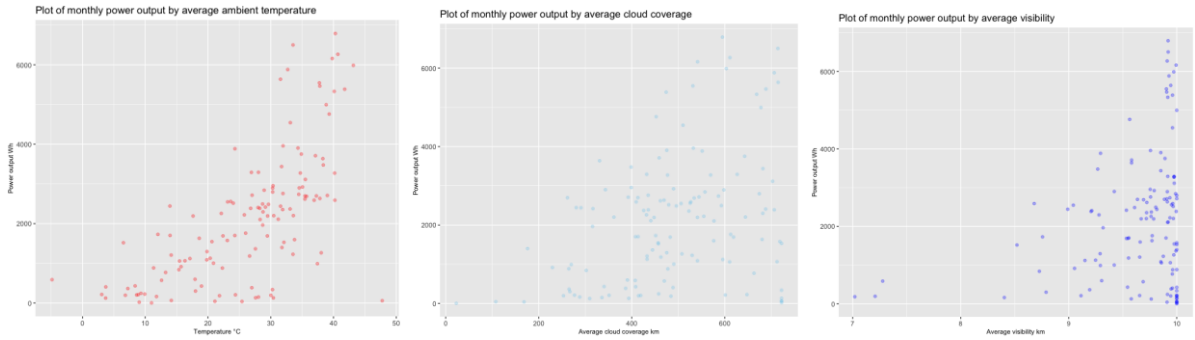
```
> summary(df) #Display statistical informations about data
   location            month          humidity          temp          wind_speed      visibility       pressure       power_output      cloud_coverage  is_enough_power_produced
 Length:128         Min.   : 1.000   Min.   : 5.92   Min.   :-4.846   Min.   : 2.547   Min.   : 7.021   Min.   : 792.0   Min.   :   3.695   Min.   : 22.8   Mode :logical
 Class :character   1st Qu.: 4.000   1st Qu.:27.09   1st Qu.:18.875   1st Qu.: 8.020   1st Qu.: 9.556   1st Qu.: 859.6   1st Qu.: 870.732   1st Qu.:408.0   FALSE:36
 Mode  :character   Median : 6.000   Median :39.98   Median :28.279   Median :10.136   Median : 9.867   Median : 982.5   Median :2151.335   Median :478.5   TRUE :92
                    Mean   : 6.469   Mean   :40.49   Mean   :26.152   Mean   :10.665   Mean   : 9.653   Mean   : 940.8   Mean   :2133.862   Mean   :488.4
                    3rd Qu.: 9.000   3rd Qu.:53.25   3rd Qu.:33.925   3rd Qu.:12.635   3rd Qu.: 9.964   3rd Qu.:1009.6   3rd Qu.:2808.265   3rd Qu.:598.7
                    Max.   :12.000   Max.   :86.60   Max.   :47.763   Max.   :27.400   Max.   :10.000   Max.   :1023.8   Max.   :6789.825   Max.   :722.0
```

Slika 15: Prikaz statističkih parametara našeg skupa podataka

U nastavku ćemo primeniti multivarijantnu analizu nad atributima koji su prikazani na histogramima. Nakon primene koda sa slike 16 vidimo redom grafikone koji pokazuju odnos temperature (identičan grafikon kao i na slici 12), prosečne pokrivenosti oblacima i prosečene vidljivosti. Ova analiza nam pokazuje da se optimalna proizvodnja električne energije, na osnovu našeg skupa podataka, postiže pri prosečnoj mesečnoj temperature između 25°C i 45°C. Takođe ne možemo da utvrdimo jasan šablon između prosečne pokrivenosti oblaka i proizvodnje električne energije, a vidimo i da prosečna vidljivost nema nikakav uticaj na proizvodnju.

```
plot_temp = ggplot(df) + geom_point(aes(temp, power_output),colour = "red", alpha =0.3) +
theme(axis.title = element_text(size = 8.5)) + ggtitle(
"Plot of monthly power output by average ambient temperature") + xlab("Temperature °C") +
ylab("Power output Wh")
plot_temp

plot_cloud_coverage = ggplot(df)+geom_point(aes(cloud_coverage, power_output),colour =
"skyblue", alpha =0.3) + theme(axis.title = element_text(size = 8.5))  + ggtitle(
"Plot of monthly power output by average cloud coverage") + xlab(
"Average cloud coverage km") + ylab("Power output Wh")
plot_cloud_coverage

plot_visibility = ggplot(df)+geom_point(aes(visibility, power_output),colour = "blue",
alpha =0.3) + theme(axis.title = element_text(size = 8.5)) + ggtitle(
"Plot of monthly power output by average visibility") + xlab("Average visibility km") +
ylab("Power output Wh")
plot_visibility
```

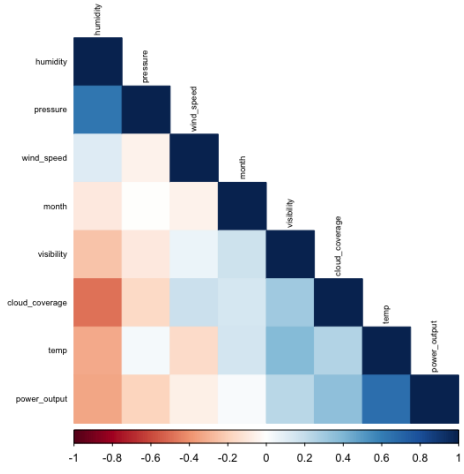
Slika 16: Kod za iscrtavanje grafikona prilikom primene multivarijantne analize



Slike 17-19: Prikaz grafikona multivarijantne analize

Kako vidimo na slici 21 nije prevelik broj blokova sa jarkim bojama pa nije potrebno primeniti PCA (eng. principal component analysis) statistički postupak, ali ćemo ipak primeniti Boruta algoritam za izbor karakteristika značajnih za primenu algoritma za mašinsko učenje.

```
df_numeric_only <- df %>% select_if(is.numeric)
#For this matrix we need to only use numerical attributes

corMatrix <- cor(df_numeric_only)
#Compute all correlations between each other and form correlation matrix.
corrplot(corMatrix,order = "FPC",method = "color",type = "lower", tl.cex
 = 0.6, tl.col = "black")
```



Slike 20 i 21: Kod za iscrtavanje i prikaz korelacione matrice

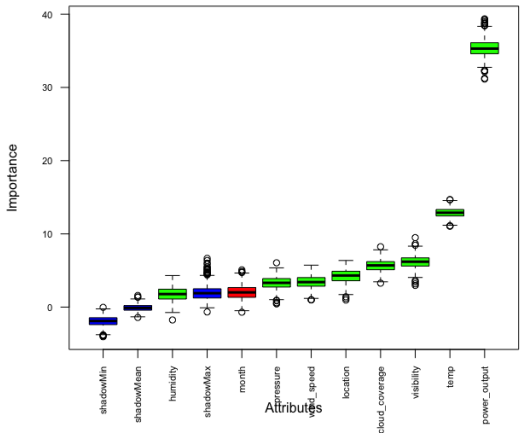
Boruta algoritam spada u tip omotačkih metoda (eng. wrapper methods) jer je stvoren oko klasifikacionog algoritma slučajnih stabala, koji ćemo i primeniti na kraju ovog rada. Boruta pokušava da obuhvati sve važne, zanimljive karakteristike koje možda imate u svom skupu podataka u odnosu na promenljivu ishoda. Algoritam funkcioniše na sledeći način:

- Prvo, on dodaje slučajnost datom skupu podataka kreiranjem izmešanih kopija svih karakteristika koje se nazivaju „Shadow Features".
- Zatim se podaci obučavju korišćenjem "Random Forest" klasifikatora na ovom proširenom skupu podataka (izvorni atributi i „Shadow Features") i primenjuje se mera važnosti obeležja poput srednja tačnost smanjenja (eng. *Mean Decrease Accuracy*) i procenjuje važnost svake karakteristike.
- Na svakoj iteraciji, Boruta algoritam proverava da li stvarna karakteristika ima veći značaj i stalno uklanja karakteristike koje se smatraju veoma nevažnim.
- Konačno, Boruta algoritam se zaustavlja ili kada se sve karakteristike potvrde ili odbiju ili kada dostigne postavljenu granicu, koja je u našem slučaju 500 iteracija.

```
boruta <- Boruta(is_enough_power_produced ~ ., data = df, doTrace = 2, maxRuns =
500) #Using Boruta for feature selection
print(boruta)
#Here we see 7 confirmed important and 2 tentative important features*
#*Note: Result may vary on each run of Boruta

boruta <- TentativeRoughFix(boruta) #We need now to check tentative features again
print(boruta)
#Now we se end result of our features and we can remove any non important feature,
again this result may vary on each run of Boruta

plot(boruta, las = 2, cex.axis = 0.7)
attStats(boruta)

getSelectedAttributes(boruta, withTentative = F)
```

Slika 22: Kod za primenu Boruta algoritma

```
> boruta <- Boruta(is_enough_power_produced ~ ., data = df, doTrace = 2, maxRuns = 500) #Using Boruta for feature selection
 1. run of importance source...
 2. run of importance source...
 3. run of importance source...
 4. run of importance source...
 5. run of importance source...
 6. run of importance source...
 7. run of importance source...
 8. run of importance source...
 9. run of importance source...
 10. run of importance source...
After 10 iterations, +0.36 secs:
 confirmed 5 attributes: cloud_coverage, location, power_output, temp, visibility;
 still have 4 attributes left.
 30. run of importance source...
After 30 iterations, +1.2 secs:
 confirmed 2 attributes: pressure, wind_speed;
 still have 2 attributes left.
 499. run of importance source...
> print(boruta) #Here we see 7 confirmed important and 2 tentative important features*
Boruta performed 499 iterations in 21.10912 secs.
 7 attributes confirmed important: cloud_coverage, location, power_output, pressure, temp and 2 more;
 No attributes deemed unimportant.
 2 tentative attributes left: humidity, month;
> boruta <- TentativeRoughFix(boruta) #We need now to check tentative features again
> print(boruta) #Now we see end result of our features and we can remove any non important feature, again this result may vary on each run of Boruta
Boruta performed 499 iterations in 21.10912 secs.
Tentatives roughfixed over the last 499 iterations.
 8 attributes confirmed important: cloud_coverage, humidity, location, power_output, pressure and 3 more;
 1 attributes confirmed unimportant: month;
> plot(boruta, las = 2, cex.axis = 0.7)
> attStats(boruta)
                meanImp medianImp     minImp    maxImp  normHits  decision
location       4.212433  4.312782  0.9651142  6.367198 0.9478958 Confirmed
month          2.0038029 2.010352 -0.6956781 5.087306 0.5170341  Rejected
humidity       1.759438  1.778897 -1.7551911  4.313015 0.4609218 Confirmed
temp          12.887337 12.888816 11.0469716 14.684126 1.0000000 Confirmed
wind_speed     3.424768  3.412486  0.9806902  5.726359 0.8517034 Confirmed
visibility     6.155632  6.198412  2.9611265  9.494365 0.9919840 Confirmed
pressure       3.282477  3.312877  0.4543389  6.049043 0.8316633 Confirmed
power_output  35.389515 35.301317 31.1431418 39.362864 1.0000000 Confirmed
cloud_coverage 5.672717  5.694990  3.2711508  8.245889 0.9879760 Confirmed
> getSelectedAttributes(boruta, withTentative = F)
[1] "location"       "humidity"       "temp"           "wind_speed"     "visibility"     "pressure"       "power_output"   "cloud_coverage"
```



Slike 23 i 24: Prikaz izvršavanja Boruta algoritma i grafikona važnosti atributa

Nakon što smo dobili važnost naših atributa utvrdili smo da možemo ukloniti mesec kao atribut, takođe ćemo ukloniti i količinu proizvedene električne energije jer ona ima direktne veze za atributom kojeg ćemo da predviđamo u nastavku, tj. da li je proizvedeno dovoljno za naše domaćinstvo. U nastavku ćemo skup podeliti na trening, test i validacioni skup, a potom izvršiti učenje algoritmom „random forest". Važno je napomenuti da se prilikom svakog poziva Bortua algoritma menja njen rezultat, tj. dolazi do promene u važnosti atributa.

4. Primena algoritma mašinskog učenja – Random forest

Pre nego što primenimo naš algoritam prvo je potrebno podeliti naš skup podataka u trening i test skup. Podela će biti izvršena u odnosu 70/30. Pored test skupa postojaće i validacioni skup u kome će tražen atribut već biti unesen, a koristiće se za proveru preciznosti našeg algoritma. U nastavku će biti prikazane slike naših skupova nakon raspodele.

```r
df <- df %>% select(-power_output)
#Attribute removed because it directly impacts is_enough_power_produced
df <- df %>% select(-month)
#Attribute removed because it is rejected by Boruta algorithm

df <- transform(
  df,
  location = as.character(location),
  humidity = as.numeric(humidity),
  temp = as.numeric(temp),
  wind_speed = as.numeric(wind_speed),
  visibility = as.numeric(visibility),
  pressure = as.numeric(pressure),
  cloud_coverage = as.numeric(cloud_coverage),
  is_enough_power_produced = as.factor(is_enough_power_produced)
)

# We'll split data into training and test sets.
set.seed(201682) #Setting seed so outcome of testing will be repeatable

sample_size = floor(0.70 * nrow(df)) #We split in proportion of 70/30
training_index = sample(seq_len(nrow(df)), size=sample_size)
training_set = subset(df[training_index,], sample = TRUE)
validation_set = subset(df[-training_index,], sample = FALSE)

test_set = validation_set %>% select(-is_enough_power_produced)

#Display first 10 rows of our sets

head(training_set)
head(validation_set)

#Display row count

dim(training_set)
dim(validation_set)
```

Slika 25: Kod za podelu skupa na trening, validacioni i test skup

```
> head(training_set)
       location humidity      temp wind_speed visibility  pressure cloud_coverage is_enough_power_produced
2672  Hill Weber 18.77000 43.159077   8.008451   9.972394  856.7758       603.2338                      TRUE
17485     Travis 51.23630 23.131548   9.647059   9.042781 1015.2070       506.0267                     FALSE
11426      MNANG 50.88133  9.312871  12.679487   9.914103  986.5346       457.8718                     FALSE
13140   Peterson 19.87390 41.780067   9.649596   9.960647  816.6261       473.9191                      TRUE
5909        JDMT 41.16074 35.712358   8.540373   9.692547 1020.1236       534.0932                     FALSE
12201      MNANG 86.59668 11.004578   7.400000  10.000000  989.6000        22.8000                     FALSE
> head(validation_set)
       location humidity     temp wind_speed visibility  pressure cloud_coverage is_enough_power_produced
140  Camp Murray 58.43724 15.71550   7.824242   9.053939 1008.6909       229.4364                     FALSE
305  Camp Murray 34.31480 24.37622   7.363636  10.000000  998.9000       351.2727                     FALSE
995  Camp Murray 29.44929 38.04071   7.733945   9.972477 1006.1211       507.8073                     FALSE
1114     Grissom 48.62353 12.60481  11.949580   9.689076  992.1899       469.9496                     FALSE
1486     Grissom 45.61178 17.94086  11.661017   9.305085  988.4237       390.6780                     FALSE
1545     Grissom 49.24025 30.36350  11.081633   9.754286  985.6567       398.9224                     FALSE
> dim(training_set)
[1] 89  8
> dim(validation_set)
[1] 39  8
```

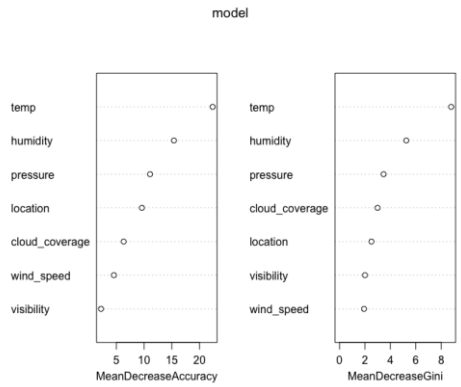
Slika 26: Izgled i količina podataka trening i validacionog skupa

```
model <- randomForest(formula = is_enough_power_produced ~ ., data
=training_set, importance=TRUE, ntree=500, type='classification')
model #Show random forest model

varImpPlot(model)
```

Slika 27: Kod za primenu „random forest“ algoritma

```
> model #Show random forest model

Call:
 randomForest(formula = is_enough_power_produced ~ ., data = training_set,      importance = TRUE, ntree = 500, type = "classification")
               Type of random forest: classification
                     Number of trees: 500
No. of variables tried at each split: 2

        OOB estimate of  error rate: 8.99%
Confusion matrix:
      FALSE TRUE class.error
FALSE    68    4  0.05555556
TRUE      4   13  0.23529412
```

Slika 28: Rezultat nakon treniranja modela



Slika 29: Grafikon važnosti atributa našeg modela

Nakon uspešnog treniranja vidimo da naš model, za koji smo koristili podrazumevani broj stabala (500) i kao tip treniranja uzeli klasifikaciju, vidimo da je OOB (eng. Out-of-bag) procena greške 12.36%, a takođe vidimo i konfuzionu matricu tačnih i netačnih klasifikacija.

Na kraju primenjujemo naš model i vidimo konfuzionu matricu predviđanja i dobijamo preciznost našeg algoritma od 82%.

```
test_set$is_enough_power_produced = predict(model, newdata=
test_set[-8]) #Assign predicted data to missing column

confusion_matrix = table(validation_set[,8], test_set[,8])
#Create confusion matrix of is_enough_power_produced values
confusion_matrix
#Display confusion matrix of is_enough_power_produced values

cat("Our model accurancy is:", mean(test_set$
is_enough_power_produced == validation_set$
is_enough_power_produced) * 100, "%")
```

```
> confusion_matrix #Display confusion matrix of is_enough_power_produced values

       FALSE TRUE
  FALSE    29    0
  TRUE      5    5
> cat("Our model accurancy is:", mean(test_set$is_enough_power_produced == validation_set$is_enough_power_produced) * 100, "%")
Our model accurancy is: 87.17949 %
```

Slike 30 i 31: Prikaz koda i rezultata testiranja našeg modela

5. Zaključak

Ovim radom smo na jednom jednostavnom primeru videli praktičnu primenu R programskog jezika za sređivanje i proveru podataka, testiranje i primenu jednog algoritma mašinskog učenja. Iako je skup podataka relativno mali, jer je period merenja samo 14 meseci, može se uspešno izvršiti analiza i predviđanje uslova da li je mesečna proizvodnja sistema veća od 1 kwh? Ukoliko bih isti skup podataka bio drastično veći moguća bi bila i primena regresije, koja bi uz manje odstupanje, predvidela planiranu količinu proizvedene električne energije na osnovu merenja za buduće mesece.