Community Contribution Report

Mert Ketenci - mk4139

3 November 2018

Indtroduction

For the community contribution we are going to establish a sorting algorithm that will sort the attributes of a given data frame according to their correlation to establish best possible visualization.

In class, we have discussed that sometimes it is hard to see certain patterns from parcoord. This happens when the data that are going to be plotted are sorted randomly.

If the sorting of the attributes are random, the trends can be hard to see.
In this community contribution we are going to inspect if different sorting methods of attributed provide a better visualization of the data.

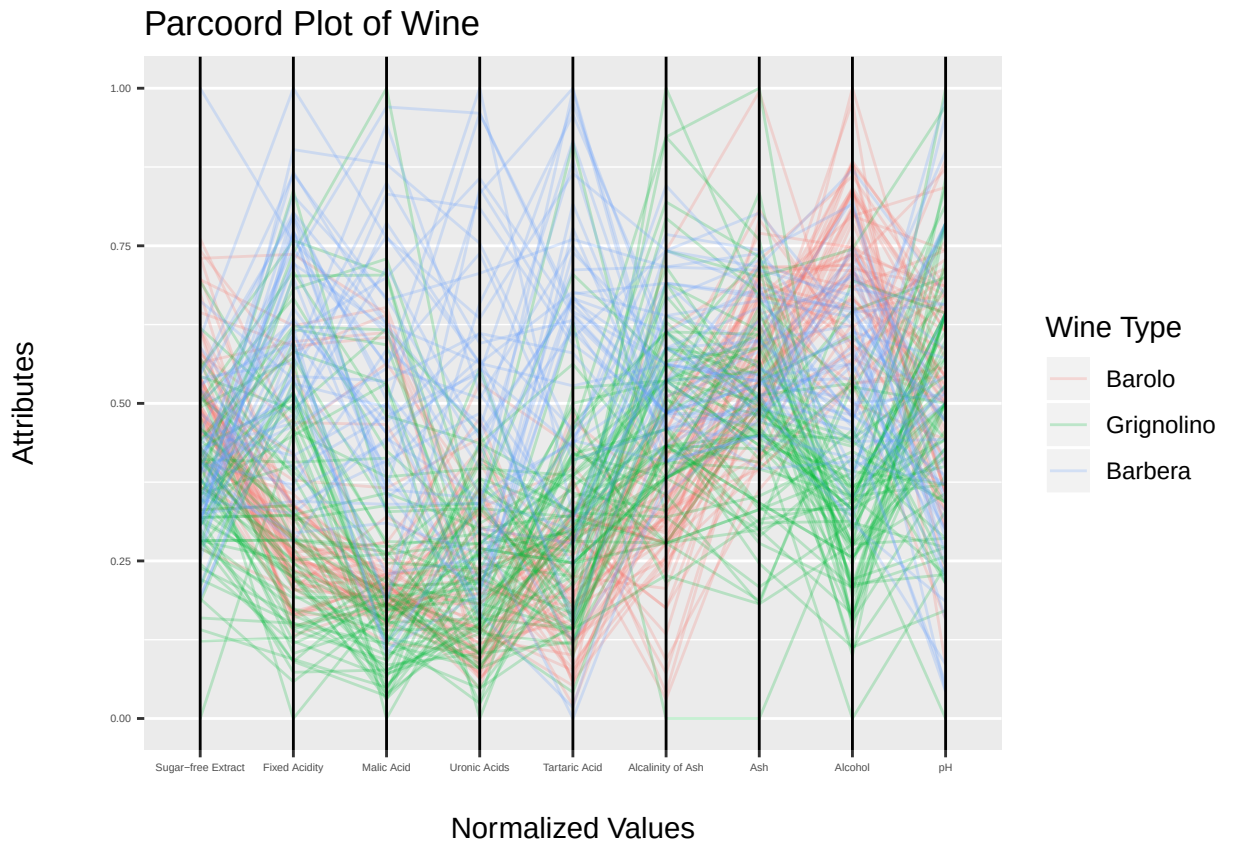
```
begining_axis="Type"

data(wine)
as.data.frame(wine)
for (i in 1:ncol(wine)){
  wine[,i]=(wine[,i]-min(wine[,i]))/(max(wine[,i])-min(wine[,i]))
}

subset_w=wine[1:10]
subset_w_shuffle=subset_w[,c("Type", "Sugar-free Extract",
                             "Fixed Acidity"
                             ,"Malic Acid", "Uronic Acids",
                             "Tartaric Acid",
                             "Alcalinity of Ash",
                             "Ash"
                             ,"Alcohol",
                             "pH")]
csubset_w=as.data.frame(cor(subset_w_shuffle)) #correlation matrix
csubset_w=csubset_w[!(row.names(csubset_w) %in% begining_axis), ]#Remove the begining axis
order=c() #An empty array to store the order
order=c(begining_axis,order)

for (i in 1:nrow(csubset_w)){

  sequence=rownames(csubset_w)[which(csubset_w==min(csubset_w[,
                                                 grep(order[i],
                                                      colnames(csubset_w))]),
                                     arr.ind=TRUE)[1]]
  order=c(order,sequence)
  row.names.remove <- rownames(csubset_w)[which(csubset_w==min(csubset_w[,
                                                          grep(order[i],
                                                               colnames(csubset_w))]),
                                                arr.ind=TRUE)[1]]

  csubset_w=csubset_w[!(row.names(csubset_w) %in% row.names.remove), ]

}
```

```
subset_w=subset_w_shuffle[,order]
```

Random Sorting

The below parcoord is from original wine[1:10] - sorted randomly. I believe in both academy and industry the data we are going to visualize is going to have its attributes randomly sorted. Thus, I randomly sorted the wine[1:10] data to observe the parcoord of a randomly sorted data. It can be seen that type 1 wines are coiled, and interpreting information from such data can be difficult.

```
subset_w_shuffle$Type <- factor(subset_w_shuffle$Type, labels = c("Barolo",
                                                                  "Grignolino",
                                                                  "Barbera"))
ggparcoord(subset_w_shuffle, columns = 2:10,
           scale = "globalminmax",
           alphaLines = .22,
           groupColumn = 1)+
  geom_vline(xintercept = 1:10)+
  theme(axis.text=element_text(size=4))+
  labs(title = "Parcoord Plot of Wine", x = "\nNormalized Values",
       y = "Attributes\n\n", color = "Wine Type")
```



Min Correlation Sorting

In the below graph the attributes are sorted according to negative correlations. **It is important to note that both plots contain same data**, but the change in visualization is worth mentioning.
The sorting is done with respect to correlation between attributes.

To examplify this:

Type attribute has strongest negative correlation with **Alcohol**.

Alcohol has strongest negative correlation with Fixed **Alcality of Ash**.

Alcality of Ash has strongest negative correlation with **Sugar Free Extract**.

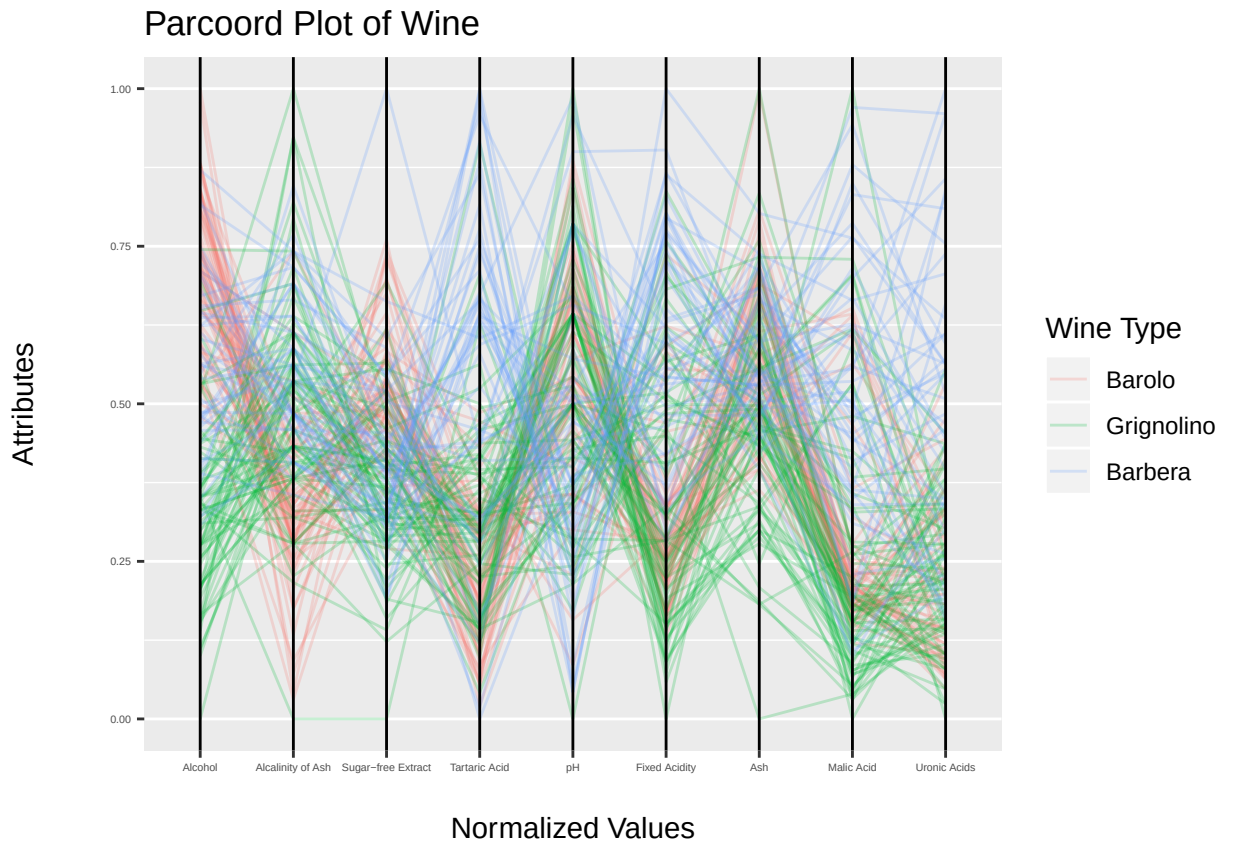
Sugar Free Extract has strongest negative correlation with **Tartaric Acid**.

and so forth.

Now we can directly say that i^{th} attribute has a negative affect on attribute $i + 1$.
So for example if a Type 0 wine has a strong Alcohol ratio, then it is certain that in the next stage (Alcality of Ash) it is going to show a weak characteristic.
If we observe the parcoord, it certainly does that.

At this point one might ask why most negative correlation. Think that we sort the attributes with respect to positive correlations. That would also certainly give us a pattern. But is I believe that it would be debatable to say that the resulting pattern would be a good visualization. Because this time the i^{th} attribute would have a positive impact on $(i-1)^{th}$ attribute. Thus, we wont observe a high frequency sinusoidal weave. Instead we would observe a continuous line for a certain attribute followed by a sudden shift.

```
subset_w$Type <- factor(subset_w$Type, labels = c("Barolo",
                                                  "Grignolino",
                                                  "Barbera"))
ggparcoord(subset_w, columns = 2:10,
           scale = "globalminmax",
           alphaLines = .22,
           groupColumn = 1)+
  geom_vline(xintercept = 1:10)+
  theme(axis.text=element_text(size=4))+
  labs(title = "Parcoord Plot of Wine", x = "\nNormalized Values",
       y = "Attributes\n\n", color = "Wine Type")
```



Working Principle of the Algorithm

1.The algorithm takes a data frame.
2.It normalizes the data (meaning that all data in each column is between 1 and 0).
3.The algorithm constructs a correlation matrix between attributes.
4.At i^{th} stage the algorithm saves the attribute that is correlated most negatively with $(i-1)^{th}$ attribute to its memory.
5.The algorithm deletes i^{th} attribute from the row of correlation data frame.
6.The algorithm moves into the next attribute.

At the end the algorithm has items that are negatively correlated with each other the most in its memory.

Data frame is sorted according to those attributes and plotted respectively.

Next Step

In the next step we are going to convert this into a package so that everyone who has trouble interpreting information from parcoord can use.

Right now, I am not knowledgeable about converting an idea into a package - thus I will do research- and would be glad to have support.

Conclusion

To conclude, It is important to indicate that the a data scientist who is going to visualize data in academy or industry might have trouble using parcoord when the attributes of the given data are sorted randomly. It also wont be wrong to state that this will be the case most of the time.

To help data scientist save time I plan to establish a simple algorithm that sorts the attributes of data with respect to correlation with another attribute.

I hope that this way data scientists are going to observe the relationship between variables through parcoord much more easily.