Advanced Python Packs

Instructor Notes, YB, RIT

Advanced Packs for Visualization and EDA

Table of Contents

Do you have messy datasets? Missing values? **missingno** provides a small toolset of flexible and easy-to-use missing data visualizations and utilities that allows you to get a quick visual summary of the completeness (or lack thereof) of your dataset.

- Visit the missingno pack site

In [ ]:
```
# import the same titanic data as in the prev workshop, and call data

df_titanic = ... # use the prev tutorials
```

missingno Pack

This pack is for misssing data. Need to read the manuals/descriptions to perform and interpret. If not comfortable, try your best or skip.

Do you have messy datasets? Missing values? **missingno** provides a small toolset of flexible and easy-to-use missing data visualizations and utilities that allows you to get a quick visual summary of the completeness (or lack thereof) of your dataset.

Visit

Let's practice some features of the pack.

In [ ]:
```
# missing data pack
import missingno as msno

# this will use the env of matplotlib
%matplotlib inline
```

In [ ]:
```
msno.matrix(df_titanic)
```

In [ ]:
```
# msno.bar is a simple visualization of nullity by column:
msno.bar(df_titanic)
```

Nullity Correlation

Nullity correlation ranges from -1 (if one variable appears the other definitely does not) to 0 (variables appearing or not appearing have no effect on one another) to 1 (if one variable appears the other definitely also does).

In [ ]:
```
# The missingno correlation heatmap measures nullity correlation: how strongly the presence or absence of one
msno.heatmap(df_titanic)
```

In [ ]:
```
# The dendrogram allows you to more fully correlate variable completion,
#revealing trends deeper than the pairwise ones visible in the correlation heatmap:
msno.dendrogram(df_titanic)
```

How to interpret Dendrogram?

- A dendrogram is a diagram that shows the hierarchical relationship between objects. It is most commonly created as an output from hierarchical clustering. The main use of a dendrogram is to work out the best way to allocate objects to clusters.
- Remember that our main interest is in similarity and clustering. Each joining (fusion) of two clusters is represented on the graph by the splitting of a horizontal line into two horizontal lines.

- To interpret this graph, read it from a top-down perspective. Cluster leaves which linked together at a distance of zero fully predict one another's presence—one variable might always be empty when another is filled, or they might always both be filled or both empty, and so on. In this specific example the dendrogram glues together the variables which are required and therefore present in every record.

ClfAutoEDA Pack for Advanced Viz

- Read the resource https://medium.com/analytics-vidhya/automated-eda-for-classification-77c25b847e43
- Download the py code in the directory you are working: https://github.com/jatinkataria94/EDA-Classification/blob/master/ClfAutoEDA.py
- Then apply features on Titanic Data.
- This pack is good at analyzing the data with the targeting variable. Pandas-profiling is not.
- Install the pack, import it and run the Auto EDA Pack with this syntax (errors may pop up). Compare the output with Pandas-profiling:

df_processed,num_features,cat_features=EDA(df_titanic_play,labels, target_variable_name, data_summary_figsize=(6,6), corr_matrix_figsize=(6,6), corr_matrix_annot=True, pairplt=True)

Sweetviz Pack

- Read the article https://towardsdatascience.com/powerful-eda-exploratory-data-analysis-in-just-two-lines-of-code-using-sweetviz-6c943d32f34
- https://towardsdatascience.com/data-frame-eda-packages-comparison-pandas-profiling-sweetviz-and-pandasgui-bbab4841943b#:~:text=more%20detailed%20information.-,Sweetviz,fully%20self%2Dcontained%20HTML%20application.
- Install it using conda (or pip) install sweetviz and run the next.
- When train and test data sets are of interest, use this pack. We will tevisit this pack once we start Machine Learning

In [ ]:
```
# import (use conda install sweetviz before this)
import sweetviz
```

In [ ]:
```
# Don't run! Read the article, use the practise of it first, Then apply on your data set.
# Initially define sata sets and label the target

df_train = ?
df_train = ?

# you can split the titanic data into train and test using train-titanic, fix this code and run the pack

#This will take too much time if
my_report = sweetviz.compare([df_train, "Train"], [df_test, "Test"], "TARGET")
```

In [ ]:
```
# Get the report
my_report.show_html("ReportXYZ.html")
# If not providing a filename, this will default to SWEETVIZ_REPORT.html
```

Pycaret Pack

A complex level pack that does all!

Read the article https://github.com/pycaret/pycaret and examples https://github.com/pycaret/pycaret/tree/master/examples.

Install it using conda (or pip) install pycaret.

Practice it with the dataset you have any or Titatnic.

In [ ]:

In [ ]:

In [ ]:

Plotly will be a separate tutorial.

Other Useful Viz and EDA Packs

- If you know any better pack for visualization in Python, please let me know so to add here. Share on the Slack
- Always first visit the main resource, not visiting the non-main pages after you google. Then practice with small data set. Then pay attention to big data and computation issues.
- Do you know any pack for big data? Share on the Slack