Week 1.2
Using Python Tools for Data Cleaning and Preprocessing

Produced By: Abdullah Abdul Majid*(linkedin)*

Completion Date: July 4, 2024

Email: abdullahpisk@gmail.com

Jupyter Notebook Links:

GitHub

Google Drive

Table of Contents:

Problem Statement:

Use Python tools to clean and prepare a dataset for further analysis.
Dataset Used: Titanic

Purpose:

The purpose of this assignment is to inspect the dataset and point out all the null values in the dataset and remove/replace them, identify and remove potential data outliers and standardize the dataset for further processing.

Process:

Dataset Loading and Initial Inspection

- Downloaded the Titanic dataset from Kaggle.
- Loaded the dataset into a pandas dataframe (eren) using *read_csv* on the train.csv file.
- To verify the dataset loading process, I used print and *head* to view the first 10 rows of the dataframe.
- Checked for *duplicated* rows, none were found.

Inspecting and Fixing the Null Values

- Used *info* to inspect the number of entries in each column and their respective datatypes

- Point out the columns with the missing data entries, they have fewer non-null entries than the total rows. In our case these are “Age”, “Cabin” and “Embarked”.
- Used *isnull* and *sum* to view the number of missing entries in the suspected columns.
- Plotted a seabnorn *heatmap* to view the missing data which was represented by the white gaps.
- To fix “Age”, I used *fillna* to fill the null data entries with the *mean* (average) of the column. This average value was chosen due to the column’s numerical nature.
- Verified the above step using *isnull* and *sum* on the “Age” column.
- To fix “Embarked”, I used *fillna* again, but this timed filled the null data entries using the *mode* (the most frequent entry) of the column. This was done since only two entries were missing and therefore the change is negligible.
- Verified again using *isnull* and *sum*.
- And to fix “Cabin”, I used *fillna* again, but this time replaced all the nulls with the string “unknown” since a significant amount of cabin numbers were missing and wrong or estimated cabin numbers might cause a problem during the processing phase.
- Verified the change again using *isnull* and *sum*.
- Confirmed all the above changes by plotting a seaborn *heatmap* again, all the gaps were filled.
- Double checked using *info*, all the null entries had been removed.

Outlier Inspection and Handling

- Plotted a seaborn *boxplot* for the numerical columns to visualize any potential outliers.
- The numerical columns were justified as follows:
    1. **PassengerId:** No outliers found; the *boxplot* was consistent.

2. Survived: No outliers found; the *boxplot* was consistent, and the output was binary.
3. Pclass: No outliers found; the *boxplot* was consistent.
4. Age: No potential outliers even though the *boxplot* is somewhat deviated, all the ages are within the usual range and the sub one ages are assumed to be infants. This column was also verified using the *min* and *max* values.
5. SibSp: No potential outliers. The Sibling-Spouse relations seem fine, even though there is a deviation in the *boxplot*, the number of relations is within the reasonable range. This was also verified using *value_counts*.
6. Parch: No potential outliers. The Parent-Child relations seem to also be fine despite the *boxplot* being massively deviated, because all the relationship counts are within reasonable range. Also verified using *value_counts*.
7. Fare: No potential outliers. Despite having a very deviated *boxplot* and very high fare spikes in the normal *plot*, the data is reasonable, and no potential outliers are found. The fare spikes are assumed to be luxury tickets. Also verified via *value_counts*.

- The categorical/object columns were justified as follows:
  1. Sex: No potential outliers. Used *value_counts* to verify. The sex was binary.
  2. Ticket: No potential outliers. Used *value_counts* to verify. There were duplicate tickets but that are assumed to be bought by a single group.
  3. Cabin: Many units unknown, other than that everything is fine.
- No potential outlier needed to be fixed and all the data is reasonable.

Data Encoding

- The categorical “Sex” and “Embarked” columns were converted to binary numerical using the one hot encoding method.

- I used the *get_dummies* pandas function for this. This split the columns up into further separate columns for each category consisting of 1s and 0s for true and false respectively.
- The above step was verified using *print* to view the first and last five rows of the dataframe.
- The objective "Cabin" and "Ticket" columns were converted to numerical integers using the label encoding method.
- I used the *factorize* pandas function for this. This gave each object a unique ordered integer. I chose this over one hot since the amount of categories was too high as there would be many columns in the data and ruin the dataframe.
- This step was also verified using *print* to view the first and last five rows of the dataframe.

Data Standardization and Transformation

- Z-Score standardization was utilized as it is less sensitive to outliers as it transforms all the columns to around a mean of 0 and gives them a standard deviation of 1.
- The module *sklearn.preprocessing* was used for *StandardScaler* function.
- The dataframe was then scaled using pandas.
- I used *print* to verify the result.