EXPLORATORY DATA ANALYSIS

few practical considerations

ABSTRACT

This document explains some of the practical aspects when we do an EDA on a given dataset. While derived metrics or features depend on a given domain, however the suggested approach in this document can be applied to all datasets from all domains.

: Shambhu Gupta

Contents

Is it Mandatory to do EDA before building ML model?

An absolute necessity! Let me highlight some of the challenges while building a ML model. A good ML model depends on the correct choice of ML algorithm and the correct data. Remember the saying – "garbage in, garbage out"! So, it is very important that we feed the highest quality of data to our ML algorithm. Some of the major challenges in building a ML model are:

- **Incomplete or Insufficient training data** – The data must be representative of all the business cases, and it should also be sufficient to reduce any possible bias.
- **Data Quality issue** – The presence of too much noise, outliers, null values, missing values, non-standardized data adversely affects the ML model accuracy. The algorithm will not be able to generalize well on poor quality data.
- **Availability of the right features/Derived Metrics** - It is very important to feed only relevant data to the ML algorithm. Too many features or too less features create problems. There is a trade-off between the model accuracy and the model complexity, and we must balance them.

What is EDA?

It can be defined as the process to explore or investigate the given dataset and summarize the main characteristics w.r.to the business problems at hand. This is mostly done with the help of visualization. Imagine the below scenarios and try to answer yourself:

- Suppose you want to study the performance of a class student in one subject. What if the dataset is only for male students? Don't you think that your study will be biased and will not be a true representation of the class as a whole?
- You sold 1000 quantity of a product, but revenue of that product is negative. Is that possible?
- Can the quantity sold be negative or can salary be negative?
- If a customer rating is allowed between 1-5 scale, what do you do when you see a rating of 7?
- How do you read the date 05/09/2022 – 5th September or 9th May?

By now, you must have started feeling the importance of EDA before building a ML model. EDA helps you to understand the data at hand and to address the data quality issue. In nutshell, EDA helps to identify (and minimize) the so called "**garbage content**" from the data. It helps us to answer two very important basic questions:

- **Correctness**: Is the data in the right shape or the right form to answer my business questions?
- **Completeness**: Does it capture all key events/ instances for our use case, or is it biased at source itself?

Lastly, EDA must be done keeping both domain and business problems in mind. A temperature of 50°C for an Indian city might look normal, whereas it might be an outlier for a European city. However, a temperature recording of 75°C for any city around the world should be flagged as erroneous.

Step by Step: Practical Consideration while performing EDA:

I. Data Understanding – Structural and Quality Investigation

Step 1: Check Completeness of Data

Often, incomplete data leads to an incomplete analysis and ultimately wrong or incorrect inferences. One must ensure that that data is complete w.r.to the business problem. For example, if you are trying to understand the customer behaviour of a global company, it may not be enough if you have data only from a specific region, say APJ alone. You should carefully analyse all the relevant data sources and integrate the data as per your business needs. Generally, the data will be stored in multiple tables or spread across multiple files/documents.

Step 2: Inspect Your Data

In this step, you import the data into python and try to inspect it at a high level. Python library ***pandas*** and ***numpy*** can be used here:

- Import your data [pd.read_csv()]
- Check data size/shape [df.shape]
- Check columns and data type [df.info()]
  - You should already have an idea to change the datatype of certain columns based on the content.
- Check the data distribution [df.describe()]
  - If you see the max values is greater than the mean + 3* Std. deviation, you should be ready to remove the outliers later.
- Check some sample records [df.head().append(df.tail())]
  - This will highlight the first and the last 5 records from the datasets. It will also hint if some header or footer is present in the dataset.

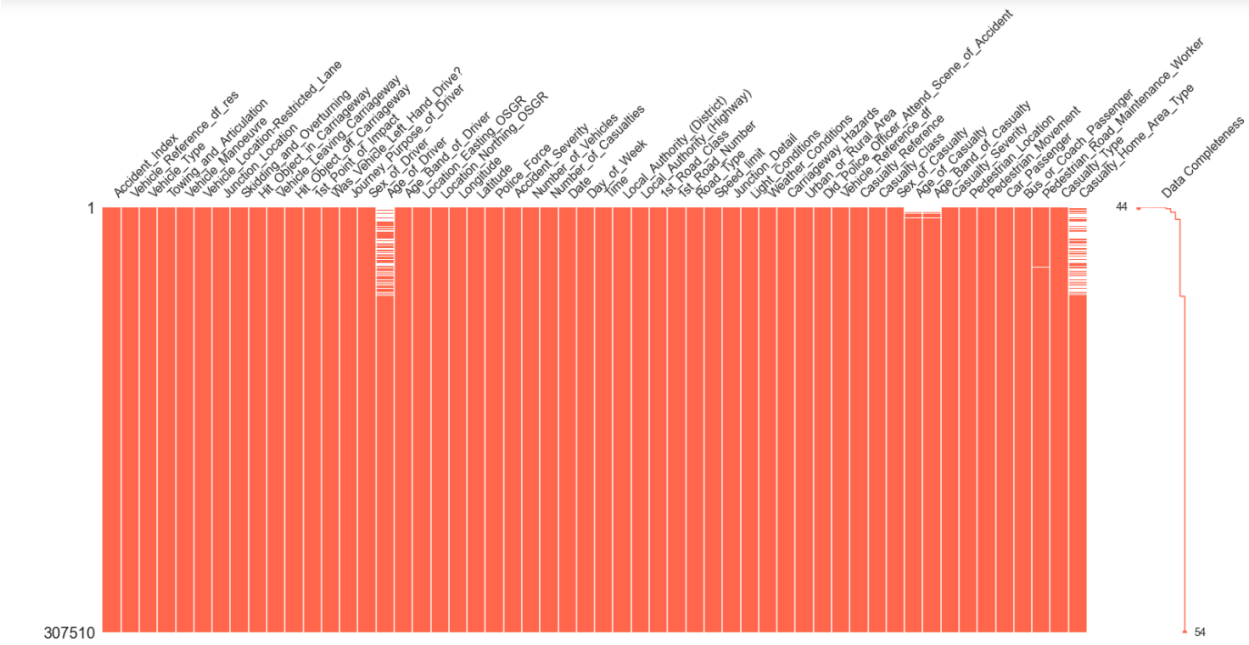
Step 3: Keep what makes sense – delete irrelevant data

- Headers/footers/page number/signatures etc... - should be deleted
- Any column that has the same values for each row (for example: Gender is all Male, or Region is APJ for all records) – should be deleted. It doesn't add any information from EDA perspective.
- Any column that has all distinct values (ex: Customer ID, Session ID, Employee ID, Serial Number, Index etc..) - should be deleted.
- Any columns that have more than 90% null values – should be deleted.
  - You can decide on the threshold cut-off based on your problem or domain specific knowledge. But any columns having more than 90% null doesn't add much info w.r.to EDA.

➢ df.isnull.sum() – will highlights number of null values per column

➢ round(df.isnull().sum()/len(df.index), 3)*100 – gives percentage of null values per column

Library **missingno** has a nice feature where you can see the distribution of NaN values in a matrix representation.



Step 4: Standardize the data

The data within a column should have a standard representation i.e., it should be on a common scale. Some of the points you can think of:

- UoM – all data must be represented using the same UoM. Ex: KMPH vs MPH, KG vs LBS, CM vs MM etc.
- Currency – ensure that currency figures are the same. USD vs EUR etc.
- Date/Timestamp – should have a common representation using the same time zone.
- Data Scale/Decimal Notation – Ensure that you have the same representation for a thousand separators (at times comma (,) or dot(.) is used as separator)

Step 5: Missing Values Treatment:

The best way to impute the missing value is that you populate the data from a reliable source. However, there may not be additional data, or it may be too time consuming. In such situation, we depend on the statistical method to impute the null or missing information. Some of the methods that can be used:

- Use mean or median for numerical column and Mode for categorical column
- Use the values from the preceding record
- Derive the values from an existing feature (such as area name from zip code, or day from date)
- Impute with a regression line

We can use classes like SimpleImputer or IterativeImputer or KNNImputer from library sklearn to do the task.

Caution: If the feature is numeric, do not blindly use mean or median to impute the values. There is a chance that the values might be encoded as numeric but represent a categorical variable. For example, vehicle type is encoded as numeric values (1 to 98) but it is actually a categorical variable. In such cases, we must use mode to impute the null values. In fact, the mean of this column might be a floating number and might not represent any vehicle type. Using mean instead of mode as imputer in such cases will result in data inconsistency. You can refer to the notebook here for this example.

Step 5: Remove Outliers

- Outliers must be removed as they distort the general behaviour of the data, and they are nonrepresentative data points. That means they do not generalize the data and it can be considered as noise.
- The best way to identify the outliers is using the box plot in python with the help of the library **matplotlib**.
- To remove the outliers, you can either use IQR method (-1.5*IQR to + 1.5*IQR) or Z-score method (-3*standard deviation to + 3*standard deviation). IQR method roughly deletes (-2.72*standard deviation to + 2.72 standard deviation) of data.

Step 6: Code/De-code your data

There may be a scenario where the data content might be coded as some values. For example, in road safety data, vehicle type is coded between 1 – 98. Or the day might be coded as between 0-6. There could be some argument that 0 represents a Sunday or Monday. The visualization of the data with these coded values might not look great as well. In such a situation, it is advisable to code/de-code the relevant values.

- Change the column name to a more meaningful name (or to rectify and spelling mistakes)
- Data content may be de-coded to represent meaningful values. (e.g. replace the values 0-6 by day of week)
- At times, you might want to encode the values so that you have a more concise visualization, especially when the values are a long text but take only a handful of values. For ex: different weather:
  1-Clear, Few clouds, Partly cloudy, Partly cloudy
  2-Mist + Cloudy, Mist + Broken clouds, Mist + Few clouds, Mist
  3-Light Snow, Light Rain + Thunderstorm + Scattered clouds, Light Rain + Scattered clouds
  4-Heavy Rain + Ice Pallets + Thunderstorm + Mist, Snow + Fog

Step 7: Perform the Final Sanity Check

In this step, domain knowledge and understanding of business problems is very important. The sanity check should be performed keeping the above two points in mind. I will try to give some examples here:

- As older transactions are not business critical, my task is to analyse data for the last 2 years only. Somehow, I see some historical data as well. In this case, I'll delete any data beyond 2 years.
- I know that my customer rating can be in the range 1-5 only (5 being the highest), so I will delete any data where the rating is not in this range. This can also be flagged as a data quality issue to the relevant team.
- Any discounts cannot be more than 100%.
- There might be certain restrictions on how the internal/external customer data is represented. Please ensure that you have the right encoding there.

II. Data Content Analysis

In the first section, we did **Structural investigation** and **Quality investigation** into our dataset. Once the structure and quality of the dataset is understood and fixed, we move to the data content analysis phase where we will analyze the data content in detail. Ideally, we should have some questions to be asked upfront based on the problem statement. I'll take an example dataset here, and try to frame some questions for data content analysis. But before that, let's understand how to analyze different features in the dataset.

The most important point in this step is to identify the dependent variables. The content should be analyzed around the dependent variables and how it relates to other features in the dataset - numerical or categorical.

Univariate Analysis:

- Numerical Feature: plots histograms [df[col].plot(kind = 'hist')]
- Categorical Feature: plot bar graphs [df[col].plot(kind = 'bar')]

Multivariate Analysis:

- Numerical vs Numerical: sns.pairplot() or sns.heatmap()
- Categorical vs Categorical: sns.countplot(hue=...)
- Categorical vs Numerical: sns.boxplot() or sns.pairplot(hue=...)

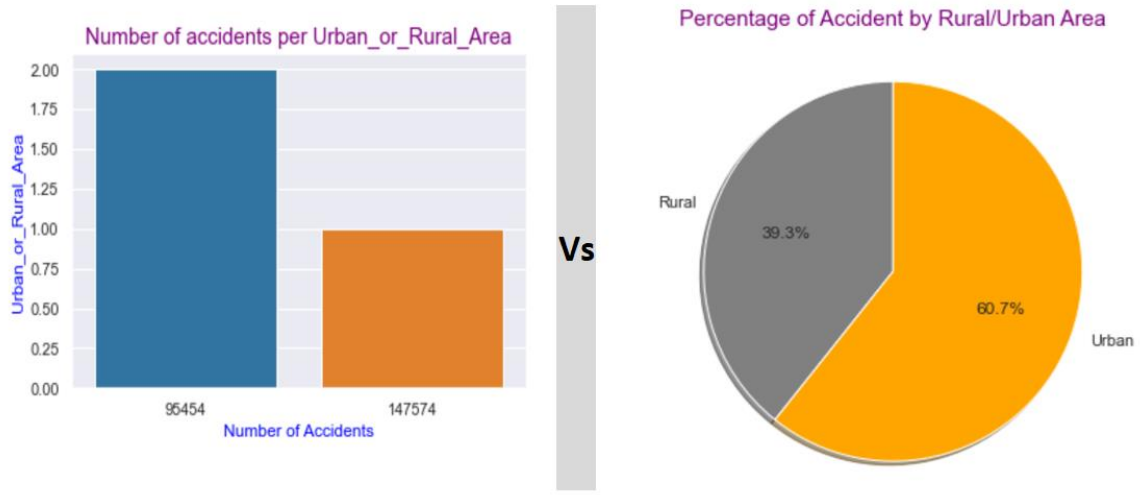
To demonstrate the idea behind the Data Content Analysis, let's consider the Road Safety Data available at Open ML. This is about the data reported to the police about the circumstances of personal injury road accidents in Great Britain from 1979, and the maker and model information of vehicles involved in the respective accident. This version includes data up to 2015. As this is data about road accidents, some of the questions that naturally come to mind are:

- How are the accident severity and number of casualties related to each other?
- Do we have more accidents on weekdays or weekends?
- Does a particular month see a greater number of accidents?
- How do causalities look round the clock?

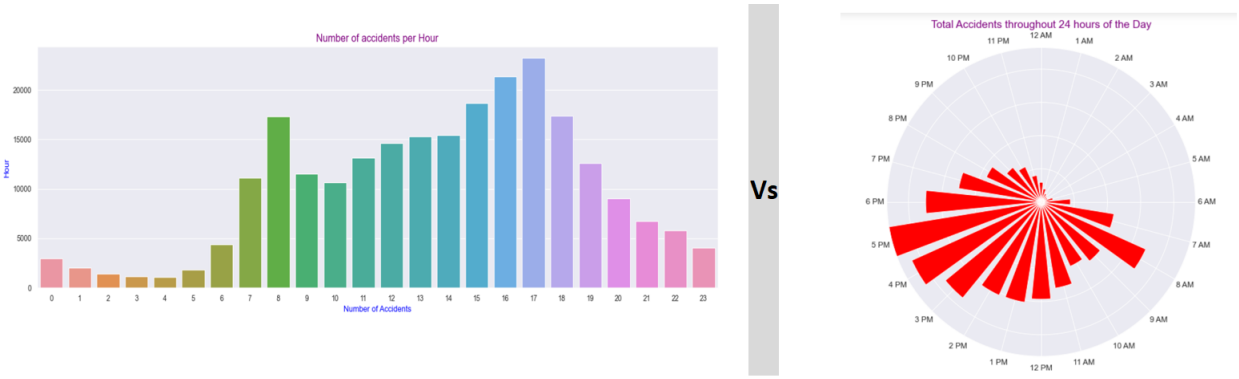
- Is there a correlation between accidents and limits?
- Is there a relationship between accidents vs road type or with junction type?
- Does the weather have any effect on accidents?
- Do we have more accidents in rural areas or in urban areas?

So, based on the problem at hand, we can start analyzing the data content to gain a deeper look into the dataset. Practically, it is not possible to analyze more than 15-20 features, so you must choose your features based on your domain knowledge.

It is also very important to choose the right visualization to depict the result. For example, if a categorical variable has only two values, you might want to use a pie chart to show the percent distribution as compared to the bar chart.



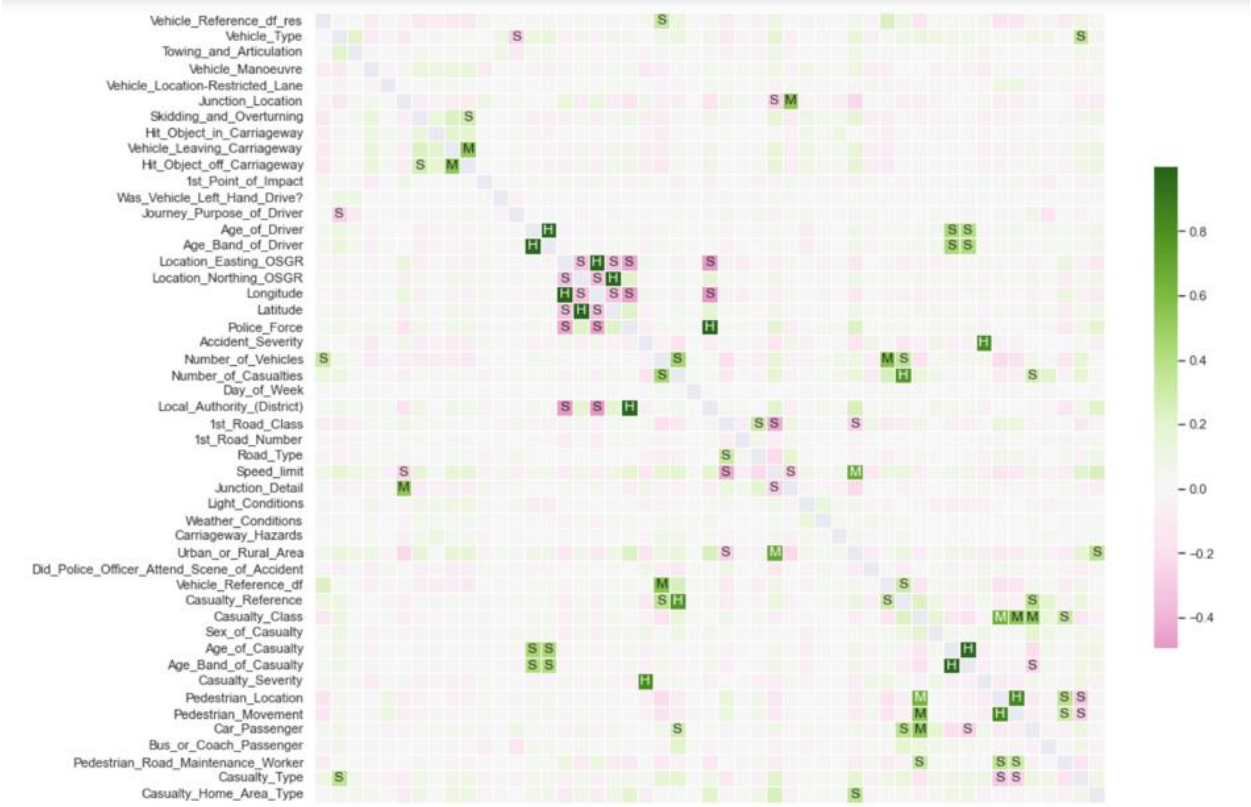
Similarly, if there is a time dimension, it would make sense to plot a radial plot rather than a simple bar chart for 24 hrs. data.



You may also want to highlight certain values or may be interested in only certain values. For example, a correlation matrix or a heatmap looks messy at times when we have more variables. A good idea would be to highlight some of the values based on correlation coefficients.

```
# Create labels for the correlation matrix
labels = np.where(np.abs(df_corr)>0.75, "H",
                  np.where(np.abs(df_corr)>0.5, "M",
                           np.where(np.abs(df_corr)>0.25, "S", "")))

# Plot correlation matrix
plt.figure(figsize=(15, 15))
sns.heatmap(df_corr, mask=np.eye(len(df_corr)), square=True,
            center=0, annot=labels, fmt='', linewidths=.5,
            cmap="PiYG", cbar_kws={"shrink": 0.5});
```



III. Derived Features/Metrics

Perhaps one of the most important and powerful tools for EDA and model building. With the right feature engineering, an average performing model can be turned into a more predictive model. However, business understanding and domain understanding is very important to do the feature engineering. For example, features like weight and height may not have any visible relationship with an ailment, but a BMI is more likely to correlate with the said ailment. Similarly, individual customers may not depict any trends, but a proper customer segmentation might have some interesting trends. In general, you can do some of the following feature engineering on your dataset:

- Try to bin some of the numeric columns into a few categories (ex: age, salary, amount, interest rate, discounts, etc…)
- Calculate various time dimensions (for example day, week, month, year etc.) from a date column.
- New feature creation like BMI from height/weight

I have come across a dataset that lists multiple solutions a customer has in his landscape. My model accuracy increased by almost 10% when I created a feature called "#Solution" and populated the values like "zero, one or two", and "more than two". I could do this as I know more solutions add to overall complexity and we should not consider individual solution names rather we should consider the number of solutions as a whole.

IV. Conclusion

In this article, we looked at some of the practical aspects that should be considered while performing an EDA. We started by investigating the dataset from a data structure and data quality perspective. We discussed some of the steps to address some of the data quality issues. Then we started analyzing the data content, its distribution, and correlation among various numerical features. However, these are certainly not all the possible content investigation and data cleaning steps one could do.

It must be noted that a proper and detailed EDA is a time-consuming process. We often say that 80% of any data science project is data preparation and EDA. However, EDA must be performed keeping the business problem in mind and exploiting the domain expertise. Please remember that a clean and relevant dataset is more likely to yield a good model. I appreciate your time and look forward to hearing your feedback.

V. Reference Notebook for Hands-on

I have included and shown some of these steps on the Road Safety Data from open ML. You can refer to the Notebook here.