A screenshot of a cell phone

Description automatically generated

A screenshot of a cell phone

Description automatically generated

**EDA** - looking for trends, outliers, exceptions, incorrect values, data anomalies

* slicing and dicing
* calculating summary statistics
* basic plotting for numerical and categorical data
* basic visualization of geospatial data on maps
* inconsistent, missing, or skewed information

1. Identification of variables and data types
2. Analyzing the basic metrics
3. Non-Graphical Univariate Analysis
4. Graphical Univariate Analysis
5. Bivariate Analysis
6. Variable transformations
7. Missing value treatment
8. Outlier treatment
9. Correlation Analysis
10. Dimensionality Reduction

* How was this data collected/where did it come from?
* Why am I interested in this data?
* What would be the target variable of interest? (if applicable)
* Is this data from a reputable source?
* Is there enough data here to make an ML model?
* Have other people conducted a similar analysis/modeling project on this dataset? Do I want to be able to learn from their conclusions or create a novel project?
* Is there a data dictionary for the dataset? Is it complete?
* Are there any additional challenges or problems that I anticipate if I use this data?
* How many features do you have?
* How many observations do you have?
* What is the data type of each feature?
* From what you know about the features of your dataset, do the data types make sense? Do you need to change any?
* Do you have null values? (to be fixed later)
* How much memory does this dataset use? Could this pose a problem for you later on?
* What is the distribution of each variable?
* Do there appear to be outliers? (to be fixed later)
* Think about what the variables mean and what the histograms say about their values and their spread — are there any surprises?

**EDA Cheat Sheet link:**  
<https://www.analyticsvidhya.com/blog/2015/06/infographic-cheat-sheet-data-exploration-python/>

**EDA considerations:**

One of the main purposes of EDA is to look at the data before assuming anything about it.

The second type of assumption, the business assumption, is a bit more elusive. With proper knowledge of a model, the data scientist knows each type of assumption that must be valid for its use and can go about systematically checking them. Business assumptions, on the other hand, can be completely unrecognized and deeply entangled with the problem and how it is framed.

**Python Techniques:**

**Common Functions:**

df.dtypes

df['Units'].count()

df.shape  
df.info()

df.describe()

df["education"].value\_counts()

distribution facet plots

df.hist(figsize=(15,15))   
#set a large figsize if you have > 9 variables  
plt.tight\_layout()  
plt.show()

df.duplicated().sum()

null = df.isna().sum()/len(df)  
null[null > 0].sort\_values()

Is the null value a result of the way data was recorded?

* Can you drop the rows with null values without it significantly affecting your analysis?
* Looking at the distributions of the variables, can you justify filling in the missing values with the mean or median for that variable?
* If your data is time-series data, can you fill the missing values with interpolation?
* Are there so many missing values for a variable that you should drop that variable from your dataset?

Outliers analysis  
  
continuous\_labels = list(continuous.columns)  
i = 1plt.figure(figsize=(15,30))for var in continuous\_labels: #plotting boxplot for each variable  
 plt.subplot(round(len(continuous\_labels),0)/3+3,4,i)  
 plt.boxplot(continuous[var],whis=5)  
 plt.title(var)  
 i+=1plt.tight\_layout()  
plt.show()

* Do you have outliers (represented as dark circles on the boxplots) in your variables?
* Why do you think you have outliers?
* Do the outliers represent real observations (i.e. not errors)?
* Should you exclude these observations? If not, should you winsorize the values?

To create a matrix of correlations for continuous variables, all the code you need is:

df.corr()

If you want to include discrete variables, then you need to use one-hot-encoding (see next section) to transform these into numeric variables. Then they can also be included in the correlation matrix.

* Which variables are most correlated with your target variable? (If applicable)
* Is there multicollinearity? (Two features that have a correlation > 0.8) How will this affect your model?
* Do you have variables that represent the same information? Can one be dropped?

|  |
| --- |
| plt.figure(figsize=(14,12)) |
|  |

|  |
| --- |
| plt.title('Pearson Correlation of Features', size = 15) |
|  |

|  |
| --- |
| colormap = sns.diverging\_palette(10, 220, as\_cmap = True) |
|  |

|  |
| --- |
| sns.heatmap(corr\_df.corr(), |
|  |

|  |
| --- |
| cmap = colormap, |
|  |

|  |
| --- |
| square = True, |
|  |

|  |
| --- |
| annot = True, |
|  |

|  |
| --- |
| linewidths=0.1,vmax=1.0, linecolor='white', |
|  |

|  |
| --- |
| annot\_kws={'fontsize':12 }) |
|  |

plt.show()

# Feature Engineering

Let’s start with a commonly used feature engineering method: variable transformation.

## Variable Transformation

The most common transformation is **one-hot-encoding** to transform categorical variables into numeric — binary, to be specific — variables. This is necessary because machine learning models cannot handle “object” data types. Pandas makes this easy to do:

new\_df = pd.get\_dummies(df,drop\_first=True)

Another common transformation (which is necessary for some models) is **standardizing** variables. Here is the code for that:

from sklearn.preprocessing import StandardScaler  
X\_std = StandardScaler().fit\_transform(X)

df.groupby(['education', 'vote']).mean()

df.corr()

Charting:

Seaborn

Matplotlib

Histogram

Correlation Heatmap

Bar Chart distributions

Scatterplots

Clustering

Whisker Plots