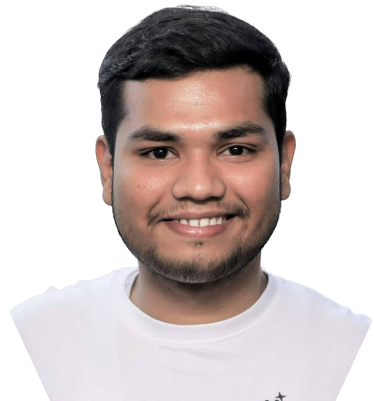


# Exploratory Data Analysis and Linear Regression

Date : 18th June 2023 | Speaker : Ayon Roy |

Event : AI/ML Month Webinar by AI Club, Indira Gandhi Delhi Technical University for Women, Delhi, India

Visit - [AYON-ROY.NETLIFY.APP](https://AYON-ROY.NETLIFY.APP)



# Hello Buddy!

I am **Ayon Roy**

**Executive Data Scientist @ NielsenIQ**

**Z by HP Global Data Science Ambassador**

Mentored/Judged **100+** Hackathons

Delivered **60+** Technical Talks

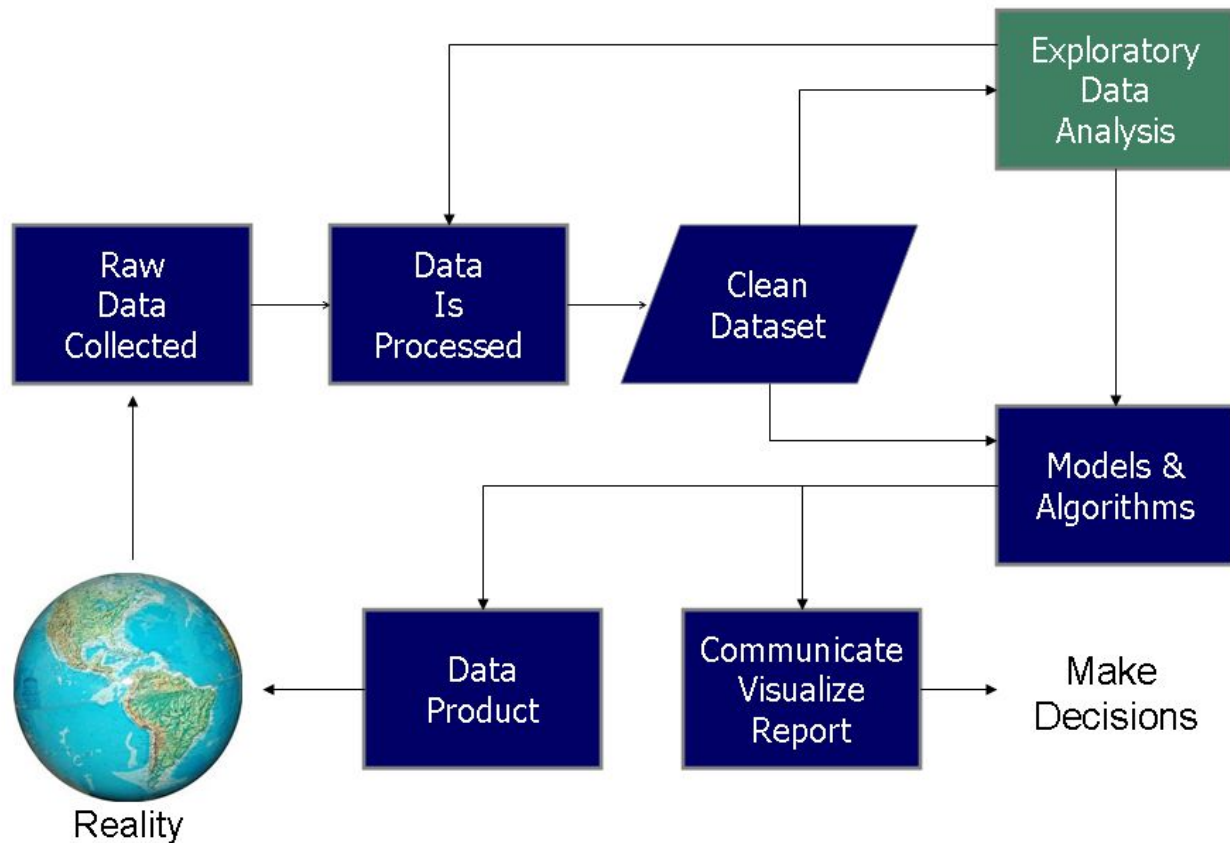
Brought **Kaggle Days Meetup** Community in India for the 1st time

**If you haven't heard about me yet, you might have been living under the rocks. Wake up !!**

# Agenda

- What is Data Pre-Processing ( DPP ) ? Why do we need it ?
- Major Steps in DPP
- What is Exploratory Data Analysis ( EDA ) ? Why do we need it ?
- What are the types of EDA ?
- Major steps in EDA
- Hands on EDA
- Linear Regression Basics
- Hands on Linear Regression

# Data Science Process



# What is Data Pre-Processing ?

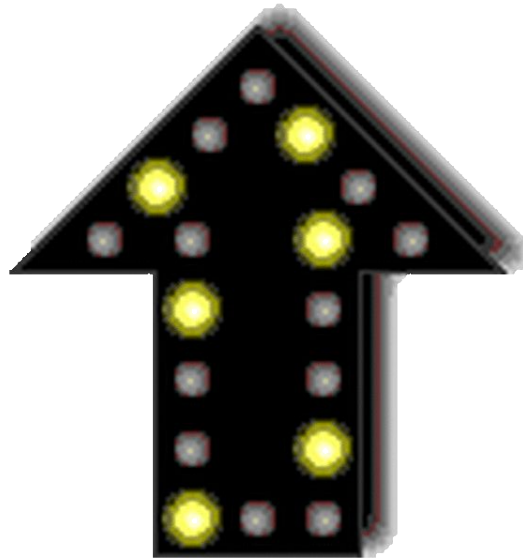
It is a technique that transforms raw data into an understandable format.

# Why do we need it ?

Raw data ( Real world data ) is always messy and that data cannot be sent through a model. That would cause certain errors.

So we need to preprocess data before sending through further analysis.

# Steps to be followed



# Get the data & Import the Libraries

```
# main libraries
import pandas as pd
import numpy as np
import time

# visual libraries
from matplotlib import pyplot as plt
import seaborn as sns
from mpl_toolkits.mplot3d import Axes3D
plt.style.use('ggplot')

# sklearn libraries
from sklearn.neighbors import KNeighborsClassifier
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import normalize
from sklearn.metrics import
confusion_matrix, accuracy_score, precision_score, recall_score, f1_score
, matthews_corrcoef, classification_report, roc_curve
from sklearn.externals import joblib
from sklearn.preprocessing import StandardScaler
from sklearn.decomposition import PCA
```



# Read the data

```
# Read the data in the CSV file using pandas
df = pd.read_csv('../input/creditcard.csv')
df.head()
```

	Time	V1	V2	V3	V4	V5	V6	V7	V8	V9	...	V21	V22	V23	V24	
0	0.0	-1.359807	-0.072781	2.536347	1.378155	-0.338321	0.462388	0.239599	0.098698	0.363787	...	-0.018307	0.277838	-0.110474	0.066928	0.12
1	0.0	1.191857	0.268151	0.166480	0.448154	0.060018	-0.082361	-0.078803	0.085102	-0.255425	...	-0.225775	-0.638872	0.101288	-0.339846	0.16
2	1.0	-1.358354	-1.340163	1.773209	0.379780	-0.503198	1.800499	0.791461	0.247676	-1.514654	...	0.247998	0.771679	0.909412	-0.689281	-0.32
3	1.0	-0.966272	-0.185226	1.792993	-0.863291	-0.010309	1.247203	0.237609	0.377436	-1.387024	...	-0.108300	0.005274	-0.190321	-1.175575	0.64
4	2.0	-1.158233	0.877737	1.548718	0.403034	-0.407193	0.095921	0.592941	-0.270533	0.817739	...	-0.009431	0.798278	-0.137458	0.141267	-0.20

Fig 1 : Dataset

# Checking the Missing Values

```
# Looking at the ST_NUM column
print df['ST_NUM']
print df['ST_NUM'].isnull()
```

Out:

```
0    104.0
1    197.0
2      NaN
3    201.0
4    203.0
5    207.0
6      NaN
7    213.0
8    215.0
```

Out:

```
0    False
1    False
2     True
3    False
4    False
5    False
6     True
7    False
8    False
```

# Remove the Missing Values

Is it a Good idea ?

Can we do something for  
this ?

# Replacing the Missing Values

---

A very common way to replace missing values is using a median.

```
# Replace using median
median = df['NUM_BEDROOMS'].median()
df['NUM_BEDROOMS'].fillna(median, inplace=True)
```

# Exploring Numerical Data & Categorical Data



Fig 2 : Class vs Frequency

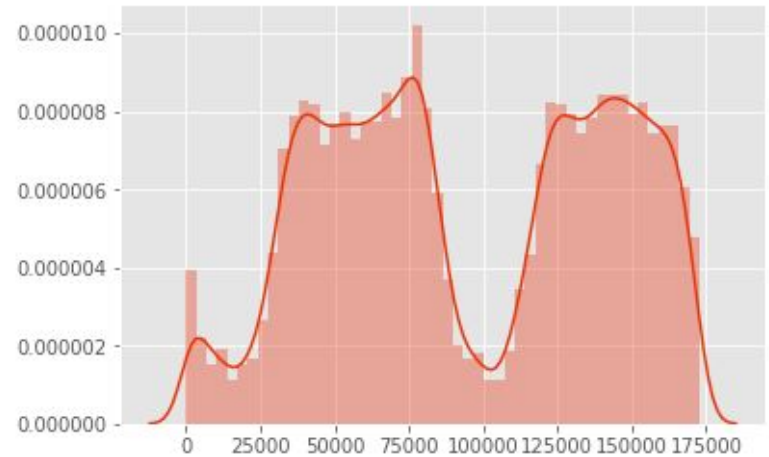


Fig 4 : Distribution of Time

# Numerical Data & Categorical Data

## Numeric

1. **Discrete** - integer values. Example: number of products bought in the shop
2. **Continuous** - any value in some admissible range (float, double).  
Example: average length of words in text

## Categorical

The variable value selected from a predefined number of categories

1. **Ordinal** - categories could be meaningfully ordered. Example: grade (A, B, C, D, E, F)
2. **Nominal** - categories don't have any order. Example: religion (Christian, Muslim, Hindu, etc.)
3. **Dichotomous/Binary** - the special case of nominal, with only 2 possible categories. Example: gender (male, female)

# Standardizing the data

```
# Standardizing the features
df['Vamount'] =
StandardScaler().fit_transform(df['Amount'].values.reshape(-1,1))
df['Vtime'] =
StandardScaler().fit_transform(df['Time'].values.reshape(-1,1))

df = df.drop(['Time','Amount'], axis = 1)
df.head()
```

V22	V23	V24	V25	V26	V27	V28	Class	Vamount	Vtime
0.277838	-0.110474	0.066928	0.128539	-0.189115	0.133558	-0.021053	0	0.244964	-1.996583
-0.638672	0.101288	-0.339846	0.167170	0.125895	-0.008983	0.014724	0	-0.342475	-1.996583
0.771679	0.909412	-0.689281	-0.327642	-0.139097	-0.055353	-0.059752	0	1.160686	-1.996562
0.005274	-0.190321	-1.175575	0.647376	-0.221929	0.062723	0.061458	0	0.140534	-1.996562
0.798278	-0.137458	0.141267	-0.206010	0.502292	0.219422	0.215153	0	-0.073403	-1.996541

Fig 7 : Standardized dataset

# Data Transformations ( Like PCA )

PCA (Principal Component Analysis) mainly using to reduce the size of the feature space while retaining as much of the information as possible.

```
X = df.drop(['Class'], axis = 1)
y = df['Class']

pca = PCA(n_components=2)
principalComponents = pca.fit_transform(X.values)
principalDf = pd.DataFrame(data = principalComponents
                           , columns = ['principal component 1', 'principal
component 2'])

finalDf = pd.concat([principalDf, y], axis = 1)
finalDf.head()
```

	principal component 1	principal component 2	Class
0	1.571633	-0.675537	0
1	-1.086136	-0.282819	0
2	2.053450	1.077546	0
3	1.150128	-0.427471	0
4	1.143864	-1.342195	0

Fig 8 : Dimensional reduction



# Data Splitting

```
# splitting the feature array and label array keeping 80% for the  
training sets  
X_train,X_test,y_train,y_test =  
train_test_split(feature_array,label_array,test_size=0.20)  
  
# normalize: Scale input vectors individually to unit norm (vector  
length).  
X_train = normalize(X_train)  
X_test=normalize(X_test)
```

# Types of Data Analysis

01

**Descriptive  
Analysis**

02

**Exploratory  
Data Analysis**

03

**Predictive  
Analysis**

04

**Inferential  
Analysis**

# Exploratory Data



# What is Exploratory Data Analysis ?

A critical process of performing initial investigations on data so as to discover patterns, to spot anomalies, to test hypothesis and to check assumptions with the help of summary statistics and graphical representations.

# Why do we need it ?

1. Detection of mistakes & missing data
2. Checking of assumptions
3. Preliminary selection of appropriate models
4. Determining relationships among the explanatory variables

**With EDA, we can make sense of the data we have and then figure out what questions we want to ask and how to frame them**

# Types of Exploratory Data Analysis

# Univariate Non-Graphical EDA

Concerned with understanding the underlying sample distribution and make observations about the population.

This also involves Outlier detection. We use the measures of central tendency like Mean, Median, Mode & measures of spread like Variance, Standard Deviation

# Univariate Graphical EDA

Histograms are used to represent the frequency with bins chosen to depict the central tendency, spread, modality, shape and outliers.

Boxplots can also be used to present information about the central tendency, symmetry and skew, as well as outliers.



# Multivariate Non-Graphical EDA

Used to show the relationship between two or more variables in the form of either cross-tabulation or statistics.

We can calculate covariance and/or correlation and assemble them into a matrix.

# Multivariate Graphical EDA

We use barplot with each group representing one level of one of the variables and each bar within a group representing the levels of the other variable.

We also use scatter plot which has one variable on the x-axis, one on the y-axis and a point for each case in your dataset. Typically, the explanatory variable goes on the X axis.

# Major Steps to be followed



# Import the Libraries

```
# Importing required libraries.  
import pandas as pd  
import numpy as np  
import seaborn as sns #visualisation  
import matplotlib.pyplot as plt #visualisation  
%matplotlib inline  
sns.set(color_codes=True)
```

# Check the type of Data

```
# Checking the data type  
df.dtypes
```

```
Make           object  
Model          object  
Year           int64  
Engine Fuel Type  object  
Engine HP      float64  
Engine Cylinders float64  
Transmission Type object  
Driven_wheels  object  
Number of Doors float64  
Market Category object  
Vehicle Size   object  
Vehicle Style  object  
highway MPG    int64  
city mpg       int64  
Popularity     int64  
MSRP           int64  
dtype: object
```

# Dropping Irrelevant Columns

```
# Dropping irrelevant columns
df = df.drop(['Engine Fuel Type', 'Market Category', 'Vehicle Style',
             'Popularity', 'Number of Doors', 'Vehicle Size'], axis=1)
df.head(5)
```

	Make	Model	Year	Engine HP	Engine Cylinders	Transmission Type	Driven_wheels	highway MPG	city mpg	MSRP
0	BMW	1 Series M	2011	335.0	6.0	MANUAL	rear wheel drive	26	19	46135
1	BMW	1 Series	2011	300.0	6.0	MANUAL	rear wheel drive	28	19	40650
2	BMW	1 Series	2011	300.0	6.0	MANUAL	rear wheel drive	28	20	36350
3	BMW	1 Series	2011	230.0	6.0	MANUAL	rear wheel drive	28	18	29450
4	BMW	1 Series	2011	230.0	6.0	MANUAL	rear wheel drive	28	18	34500

Dropping irrelevant columns.

# Renaming the Columns

```
# Renaming the column names
df = df.rename(columns={"Engine HP": "HP", "Engine Cylinders":
"Cylinders", "Transmission Type": "Transmission", "Driven_Wheels":
"Drive Mode", "highway MPG": "MPG-H", "city mpg": "MPG-C", "MSRP":
"Price" })
df.head(5)
```

	Make	Model	Year	HP	Cylinders	Transmission	Drive Mode	MPG-H	MPG-C	Price
0	BMW	1 Series M	2011	335.0	6.0	MANUAL	rear wheel drive	26	19	46135
1	BMW	1 Series	2011	300.0	6.0	MANUAL	rear wheel drive	28	19	40650
2	BMW	1 Series	2011	300.0	6.0	MANUAL	rear wheel drive	28	20	36350
3	BMW	1 Series	2011	230.0	6.0	MANUAL	rear wheel drive	28	18	29450
4	BMW	1 Series	2011	230.0	6.0	MANUAL	rear wheel drive	28	18	34500

Renaming the column name.

# Removing the Duplicates

```
# Dropping the duplicates  
df = df.drop_duplicates()  
df.head(5)
```

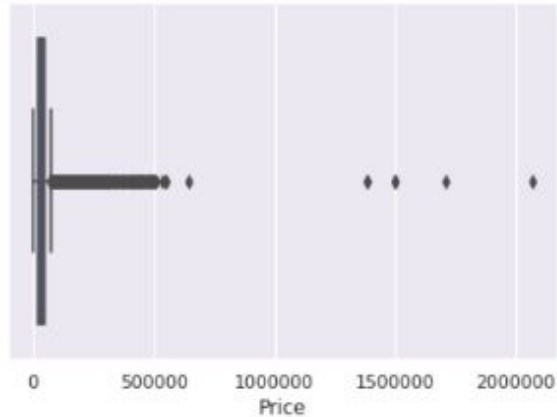
	Make	Model	Year	HP	Cylinders	Transmission	Drive Mode	MPG-H	MPG-C	Price
0	BMW	1 Series M	2011	335.0	6.0	MANUAL	rear wheel drive	26	19	46135
1	BMW	1 Series	2011	300.0	6.0	MANUAL	rear wheel drive	28	19	40650
2	BMW	1 Series	2011	300.0	6.0	MANUAL	rear wheel drive	28	20	36350
3	BMW	1 Series	2011	230.0	6.0	MANUAL	rear wheel drive	28	18	29450
4	BMW	1 Series	2011	230.0	6.0	MANUAL	rear wheel drive	28	18	34500



# Detecting the Outliers

```
sns.boxplot(x=df['Price'])
```

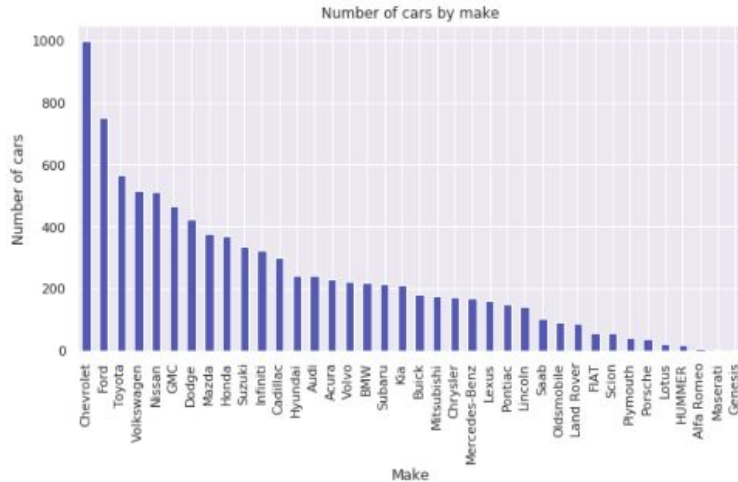
<matplotlib.axes.\_subplots.AxesSubplot at 0x7f69f68edc18>



# Plotting different features

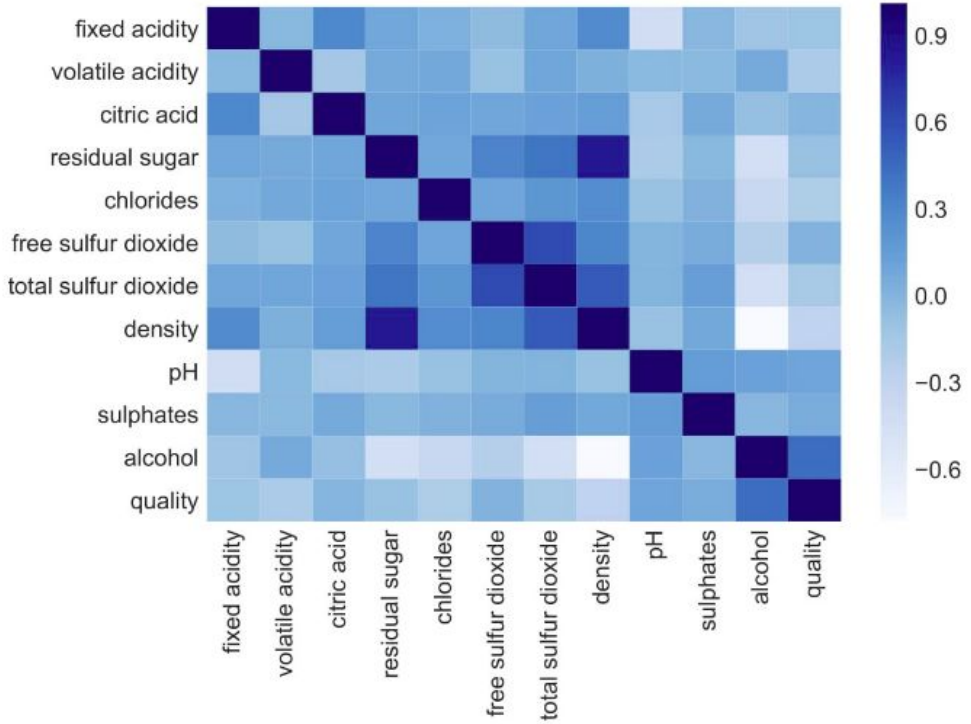
## # Plotting a Histogram

```
df.Make.value_counts().nlargest(40).plot(kind='bar', figsize=(10,5))  
plt.title("Number of cars by make")  
plt.ylabel('Number of cars')  
plt.xlabel('Make');
```



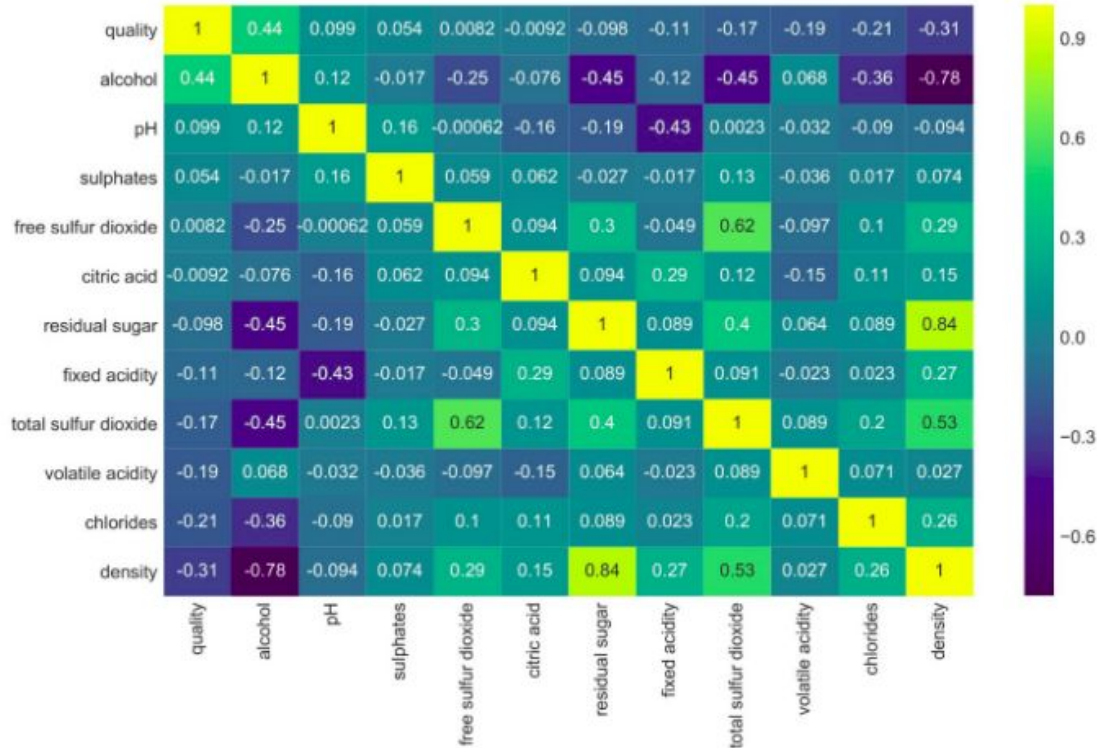
Histogram

# Heatmaps



Heatmap

# Correlation Matrix etc.



- Hands On EDA
- Linear Regression Basics
- Hands on Linear Regression



Refer to the content shared

# Danke Schoen

Questions ? Any Feedbacks ? Did you like the talk?  
Tell me about it.

If you think I can help you,  
connect with me via

Email : [ayon-roy@outlook.com](mailto:ayon-roy@outlook.com)

LinkedIn : <https://www.linkedin.com/in/ayon-roy>

Instagram : <https://www.instagram.com/iayonroy/>

Website : [AYON-ROY.NETLIFY.APP](https://AYON-ROY.NETLIFY.APP)



Scan to connect