

# Data Preprocessing



# Hello!

I am Eslam Ahmed

I am a software engineer.

You can find me at [jeksogsa@gmail.com](mailto:jeksogsa@gmail.com)



# Agenda

- hexagon icon What is Data
- hexagon icon Machine Learning
- hexagon icon Data Preprocessing
- hexagon icon Feature Engineering and Extraction
  - Domain knowledge features
  - Date and Time features
  - String operations
  - Web Data
  - Geospatial features
  - Work with Text
  - Work with Images
  - Work with Audio
  - Dimension Reduction with PCA
- hexagon icon Feature Transformations
  - Data Cleaning or Cleansing
  - Work with Missing data
  - Work with Categorical data
  - Detect and Handle Outliers
  - Split data to Train and Test Sets
  - Deal with Imbalanced classes
  - Feature Scaling
  - How to use Pipelines
- hexagon icon Feature Selection

# Agenda

## What is Data

## Machine Learning

## Data Preprocessing

## Feature Engineering and Extraction

- Domain knowledge features
- Date and Time features
- String operations
- Web Data
- Geospatial features
- Work with Text
- Work with Images
- Work with Audio
- Dimension Reduction with PCA

## Feature Transformations

- Data Cleaning or Cleansing
- Work with Missing data
- Work with Categorical data
- Detect and Handle Outliers
- Split data to Train and Test Sets
- Deal with Imbalanced classes
- Feature Scaling
- How to use Pipelines

## Feature Selection

# What is Data

We are living in a data-driven economy. It's a world where having tons of data, understanding it and knowing what to do with data is power.

Understanding your data is not one of the most difficult things in data science, but it is time-consuming.  
Interpretation of data is effective when we know about the source of data.



# What is Data (Types of Data)

## Categorical

This represents qualitative data with no apparent inherent mathematical meaning.

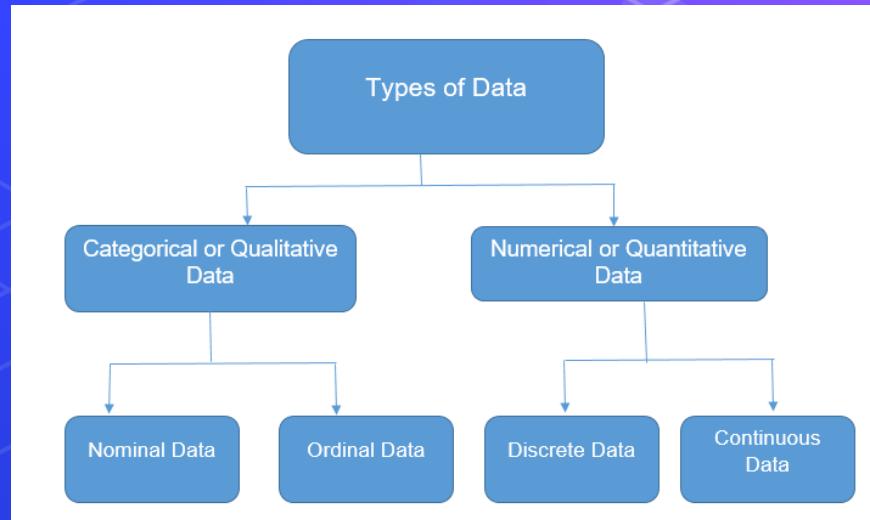
Example: Yes or No, Sex, Race, Marital Status etc.

These can be assigned numbers like Yes(0) and No(1), but numbers have no mathematical meaning.

## Numerical

This represents some sort of quantitative measurement.

Example: height of people, stock price, page load time etc



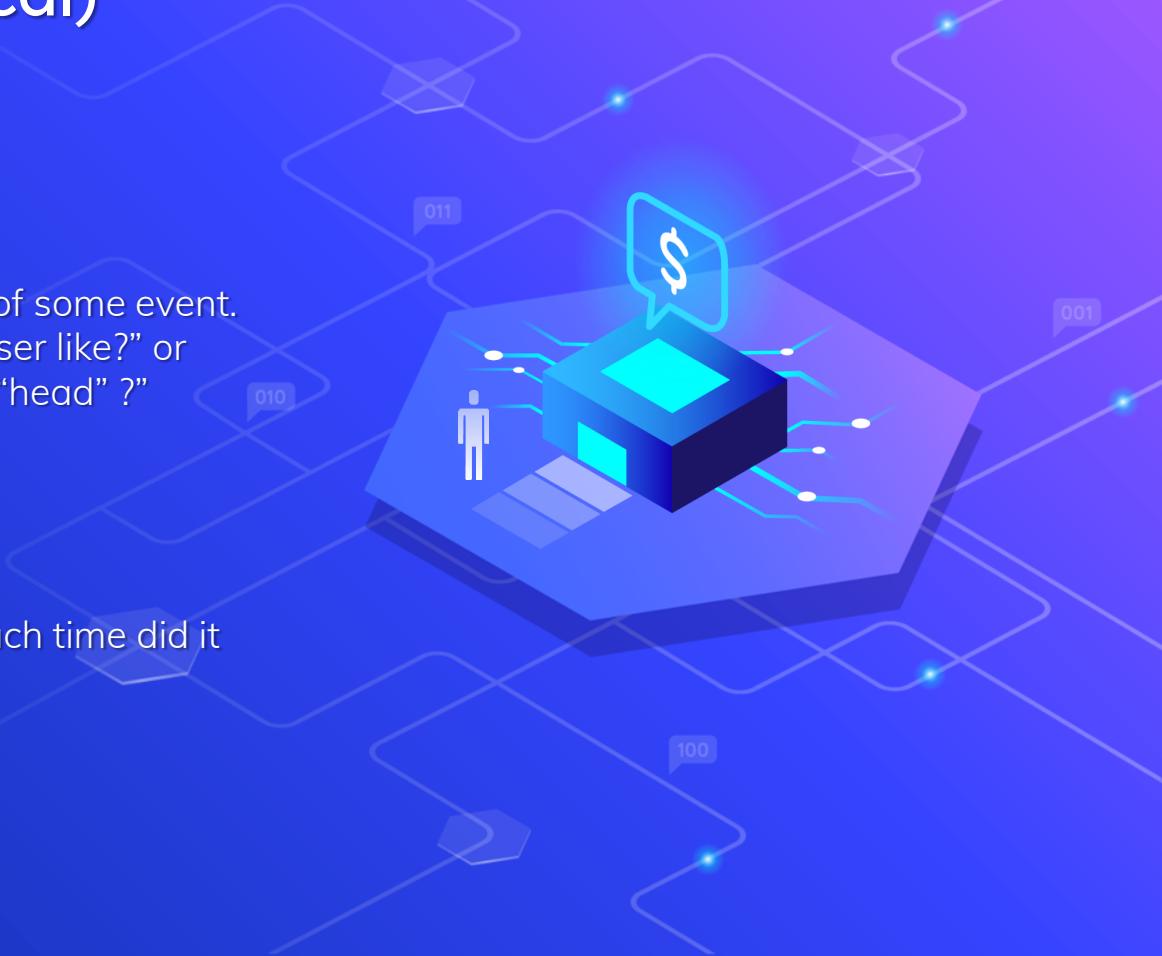
# What is Data (Numerical)

## Discrete data

This is integer based, often counts of some event.  
Example: "How many songs do a user like?" or  
"How many times a coin flipped to "head" ?"

## Continuous data

It has an infinite number of  
possible values. Example: "How much time did it  
take for a user to buy?"



# What is Data (Categorical)

## Nominal or unordered data

we assign individual items to named categories that do not have an implicit or natural value or rank.

Example: Animals can be cats or dogs or horses etc.. that would be nominal data.

## Ordered or ordinal data

items are assigned to categories that do have some kind of implicit or natural order.

Example: "Small, Medium, or Large." Another example is a survey question that asks us to rate an item on a 1 to 10 scale, with 10 being the best. This implies that 10 is better than 9, which is better than 8, and so on.

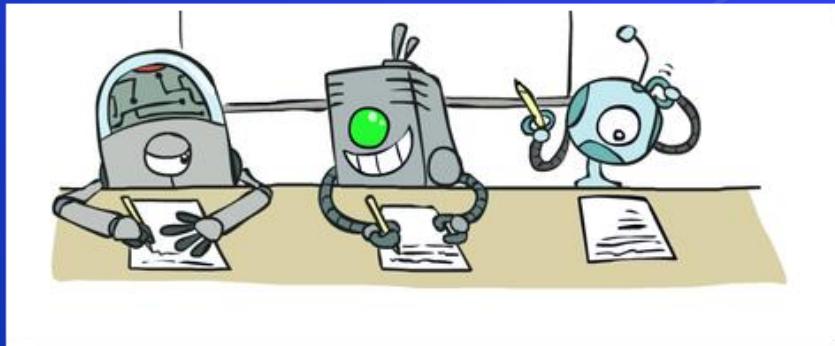


# Agenda

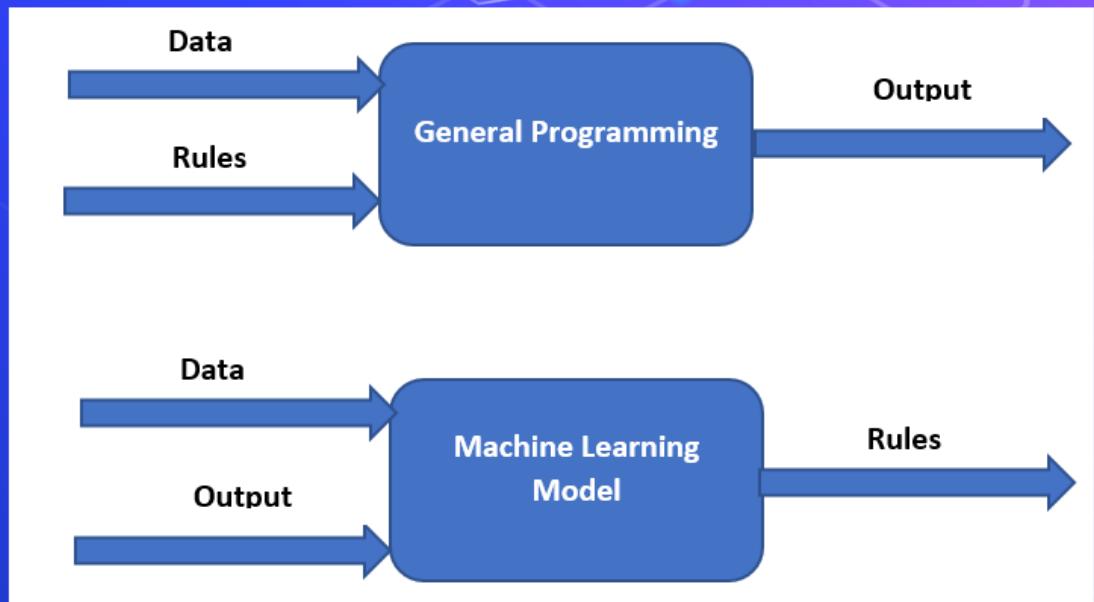
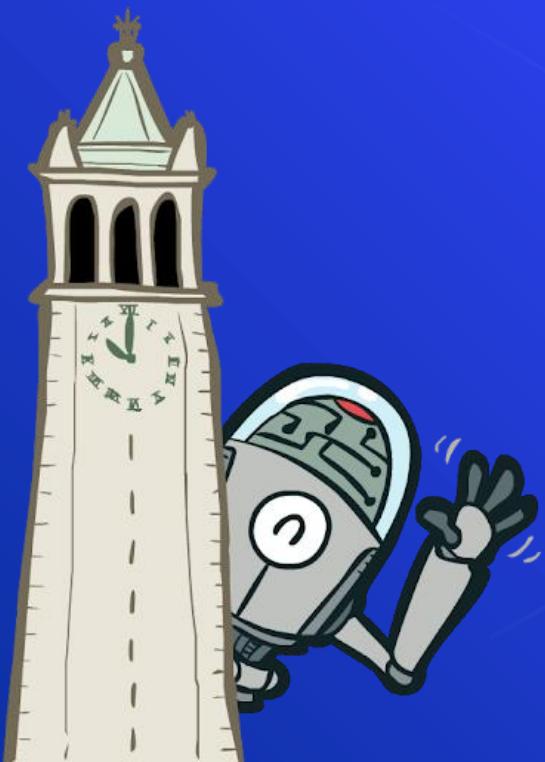
- hexagon icon What is Data
- hexagon icon **Machine Learning**
- hexagon icon Data Preprocessing
- hexagon icon Feature Engineering and Extraction
  - Domain knowledge features
  - Date and Time features
  - String operations
  - Web Data
  - Geospatial features
  - Work with Text
  - Work with Images
  - Work with Audio
  - Dimension Reduction with PCA
- hexagon icon Feature Transformations
  - Data Cleaning or Cleansing
  - Work with Missing data
  - Work with Categorical data
  - Detect and Handle Outliers
  - Split data to Train and Test Sets
  - Deal with Imbalanced classes
  - Feature Scaling
  - How to use Pipelines
- hexagon icon Feature Selection

# Machine Learning

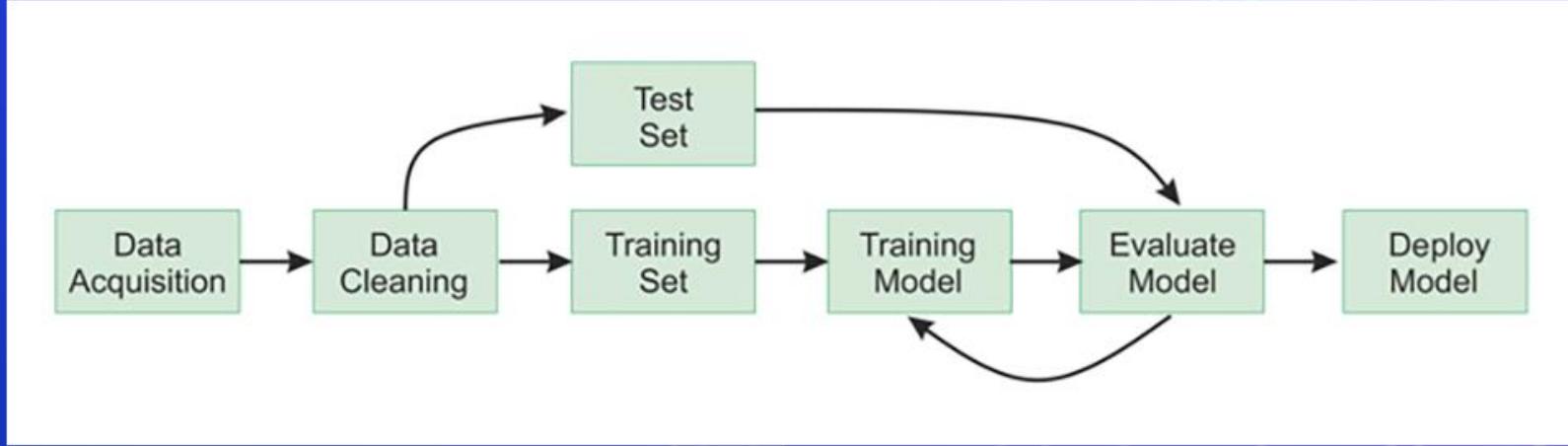
Machine learning involves computers discovering how they can perform tasks without being explicitly programmed to do so. It involves computers learning from data provided so that they carry out certain tasks.



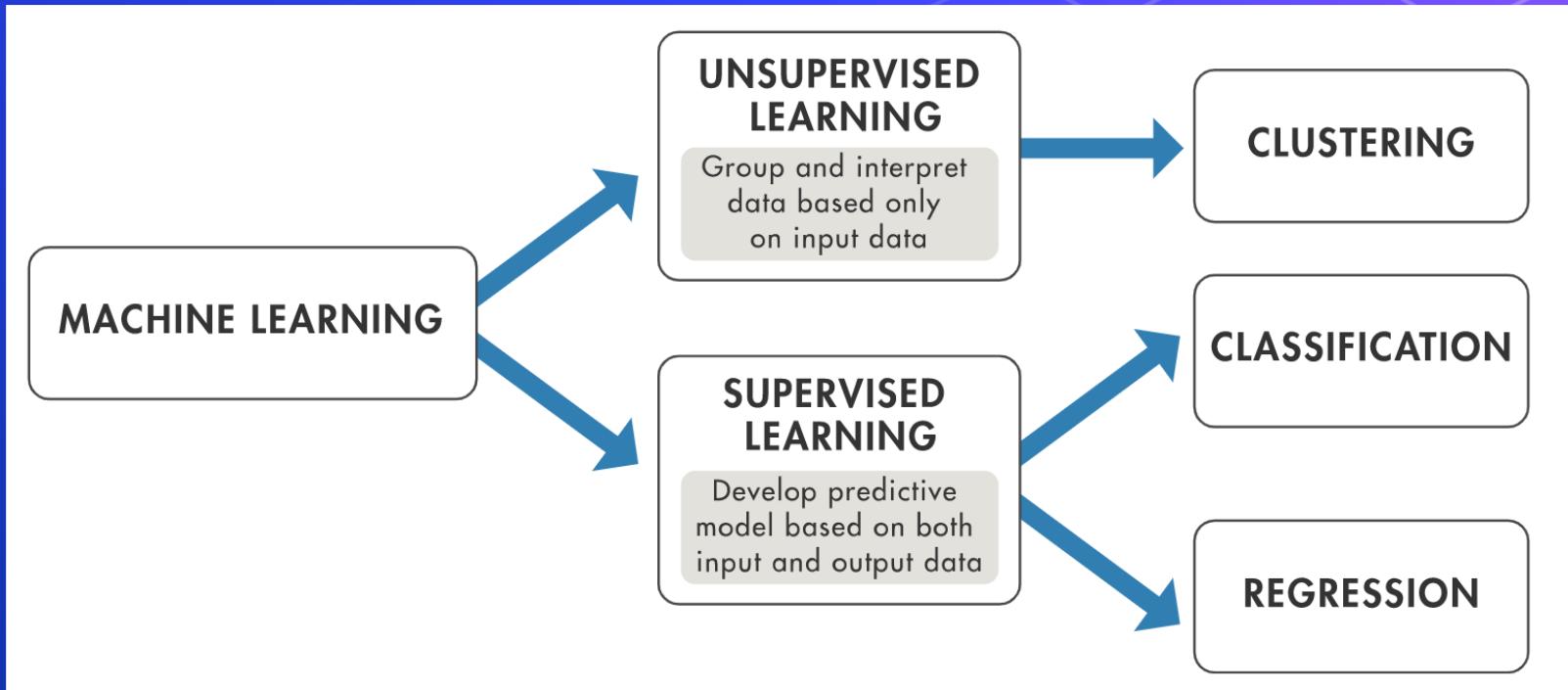
# Machine Learning



# Machine Learning



# Machine Learning



# Agenda

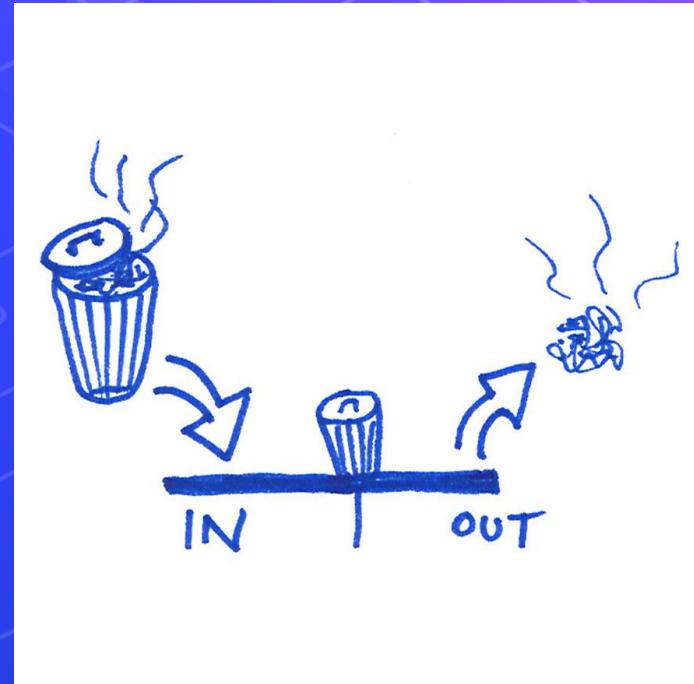
- hexagon icon What is Data
- hexagon icon Machine Learning
- hexagon icon **Data Preprocessing**
- hexagon icon Feature Engineering and Extraction
  - Domain knowledge features
  - Date and Time features
  - String operations
  - Web Data
  - Geospatial features
  - Work with Text
  - Work with Images
  - Work with Audio
  - Dimension Reduction with PCA
- hexagon icon Feature Transformations
  - Data Cleaning or Cleansing
  - Work with Missing data
  - Work with Categorical data
  - Detect and Handle Outliers
  - Split data to Train and Test Sets
  - Deal with Imbalanced classes
  - Feature Scaling
  - How to use Pipelines
- hexagon icon Feature Selection

# Data Preprocessing

The phrase "garbage in, garbage out" is the best description for the use of data preprocessing.

Data gathering methods are often loosely controlled, resulting in out-of-range values, useless features, outliers, missing values, etc.

Garbage data produce misleading results. Thus, data preprocessing is the most important phase of a machine learning project.



# Data Preprocessing

## Feature Engineering and Extraction

Extract and Create useful features from raw data into features suitable for modeling.

## Feature Transformation

Transformation of data to improve the accuracy of the algorithm.

## Feature Selection

Removing unnecessary features.



# Data Preprocessing

## Install

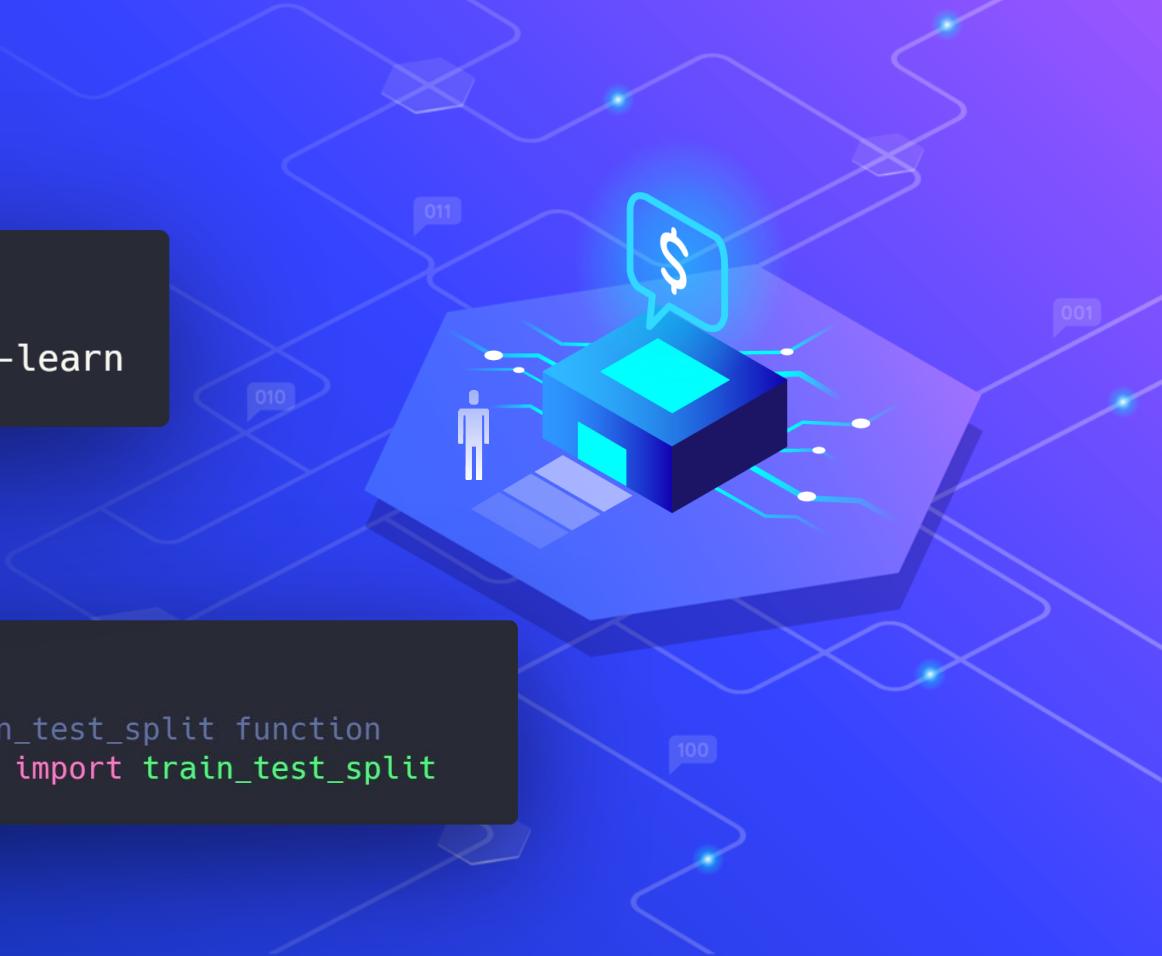


```
1 >_ conda install scikit-learn
```

## Use



```
1 # example for importing train_test_split function  
2 from sklearn.model_selection import train_test_split
```



# Agenda

- hexagon icon What is Data
- hexagon icon Machine Learning
- hexagon icon Data Preprocessing
- hexagon icon **Feature Engineering and Extraction**
  - Domain knowledge features
  - Date and Time features
  - String operations
  - Web Data
  - Geospatial features
  - Work with Text
  - Work with Images
  - Work with Audio
  - Dimension Reduction with PCA

- hexagon icon Feature Transformations
  - Data Cleaning or Cleansing
  - Work with Missing data
  - Work with Categorical data
  - Detect and Handle Outliers
  - Split data to Train and Test Sets
  - Deal with Imbalanced classes
  - Feature Scaling
  - How to use Pipelines
- hexagon icon Feature Selection

# Domain knowledge features

It's the process of creating new useful features from current features based on our understanding of the problem domain knowledge.



```
1 # example for delivery system we calculate speed in Km/m  
2 df['speed'] = df['Distance (KM)'] / (df['Time from Pickup to Arrival'] / 60)
```

# Agenda

- hexagon icon What is Data
- hexagon icon Machine Learning
- hexagon icon Data Preprocessing
- hexagon icon **Feature Engineering and Extraction**
  - Domain knowledge features
  - **Date and Time features**
  - String operations
  - Web Data
  - Geospatial features
  - Work with Text
  - Work with Images
  - Work with Audio
  - Dimension Reduction with PCA
- hexagon icon Feature Transformations
  - Data Cleaning or Cleansing
  - Work with Missing data
  - Work with Categorical data
  - Detect and Handle Outliers
  - Split data to Train and Test Sets
  - Deal with Imbalanced classes
  - Feature Scaling
  - How to use Pipelines
- hexagon icon Feature Selection

# Date and Time features

Date columns usually provide valuable information about the problem, they are neglected as an input for the machine learning algorithms. It might be the reason for this, that dates can be present in numerous formats, which make it hard to understand by algorithms.



```
1 df['Time'] = pd.to_datetime(df['Time'],format='%d/%m/%Y %H:%M')
```

# Date and Time features

```
1 df['Year'] = df['Time'].dt.year  
2 df['Month'] = df['Time'].dt.month  
3 df['Month_Name'] = df['Time'].dt.month_name()  
4 df['Week'] = df['Time'].dt.week  
5 df['Day'] = df['Time'].dt.day  
6 df['Week_Day'] = df['Time'].dt.weekday  
7 df['Day_Name'] = df['Time'].dt.day_name()  
8 df['Hour'] = df['Time'].dt.hour  
9 df['Minute'] = df['Time'].dt.minute
```

- Extract day, hour, minute, seconds, quarter, month, year, etc.
- Extract time-based features like evenings, noon, night time etc.
- Extract seasonal features like winter, summer, autumn.
- Calculate time elapsed between two related Date features.



# Date and Time features

```
1 def map_hours(x):
2     if x in [0,1,2,3,4,5,6,7,8,9,10,11,12]:
3         return 'morning'
4     elif x in [13,14,15,16]:
5         return 'afternoon'
6     else:
7         return 'evening'
8
9 df['Period'] = df['Hour'].apply(map_hours)
```

- Extract day, hour, minute, seconds, quarter, month, year, etc.
- **Extract time-based features like evenings, noon, night time etc.**
- Extract seasonal features like winter, summer, autumn.
- Calculate time elapsed between two related Date features.



# Date and Time features

```
1 def map_months(x):
2     if x in [12, 1, 2]:
3         return 'Winter'
4     elif x in [3, 4, 5]:
5         return 'Spring'
6     elif x in [6, 7, 8]:
7         return 'Summer'
8     elif x in [9, 10, 11]:
9         return 'Autumn'
10
11 df['Season'] = df['Month'].apply(map_months)
```

- Extract day, hour, minute, seconds, quarter, month, year, etc.
- Extract time-based features like evenings, noon, night time etc.
- **Extract seasonal features like winter, summer, autumn.**
- Calculate time elapsed between two related Date features.



# Date and Time features



```
1 df['Elapsed_Years'] = (datetime.now() - df['Time']) / np.timedelta64(1, 'Y')
```

- Extract day, hour, minute, seconds, quarter, month, year, etc.
- Extract time-based features like evenings, noon, night time etc.
- Extract seasonal features like winter, summer, autumn.
- Calculate time elapsed between two related Date features.

# Agenda

- hexagon icon What is Data
- hexagon icon Machine Learning
- hexagon icon Data Preprocessing
- hexagon icon **Feature Engineering and Extraction**
  - Domain knowledge features
  - Date and Time features
  - **String operations**
  - Web Data
  - Geospatial features
  - Work with Text
  - Work with Images
  - Work with Audio
  - Dimension Reduction with PCA
- hexagon icon Feature Transformations
  - Data Cleaning or Cleansing
  - Work with Missing data
  - Work with Categorical data
  - Detect and Handle Outliers
  - Split data to Train and Test Sets
  - Deal with Imbalanced classes
  - Feature Scaling
  - How to use Pipelines
- hexagon icon Feature Selection

# String operations

There is a lot of data hidden within strings, you can use a lot of string operations techniques to extract those data.

```
1 def extract_email_provider(email):
2     return email.split('@')[1]
3
4 df['Email_Provider'] = df['Email'].apply(extract_email_provider)
```

# Agenda

- hexagon icon What is Data
- hexagon icon Machine Learning
- hexagon icon Data Preprocessing
- hexagon icon **Feature Engineering and Extraction**
  - Domain knowledge features
  - Date and Time features
  - String operations
  - Web Data
  - Geospatial features
  - Work with Text
  - Work with Images
  - Work with Audio
  - Dimension Reduction with PCA

- hexagon icon Feature Transformations
  - Data Cleaning or Cleansing
  - Work with Missing data
  - Work with Categorical data
  - Detect and Handle Outliers
  - Split data to Train and Test Sets
  - Deal with Imbalanced classes
  - Feature Scaling
  - How to use Pipelines
- hexagon icon Feature Selection

# Web Data

There are a lot of web applications data logs like user agents that contain a lot of info about used browser, OS and device.

```
1 import user_agents
2
3 ua = 'Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)
4   Chrome/83.0.4103.116 Safari/537.36'
5 ua = user_agents.parse(ua)
6
7 print('Is a bot? ', ua.is_bot)
8 print('Is mobile? ', ua.is_mobile)
9 print('Is PC? ', ua.is_pc)
10 print('OS Family: ', ua.os.family)
11 print('OS Version: ', ua.os.version)
12 print('Browser Family: ', ua.browser.family)
13 print('Browser Version: ', ua.browser.version)
14 print('Device Family: ', ua.device.family)
15 print('Device Brand: ', ua.device.brand)
16 print('Device Model: ', ua.device.model)
```

# Web Data

IP address is a great way to extract geographical data from it.

```
1 from ip2geotools.databases.noncommercial import DbIpCity as ip2geo
2
3
4 response = ip2geo.get('45.243.72.231', api_key='free')
5
6 print(response.ip_address)
7 print(response.city)
8 print(response.region)
9 print(response.country)
10 print(response.latitude)
11 print(response.longitude)
```

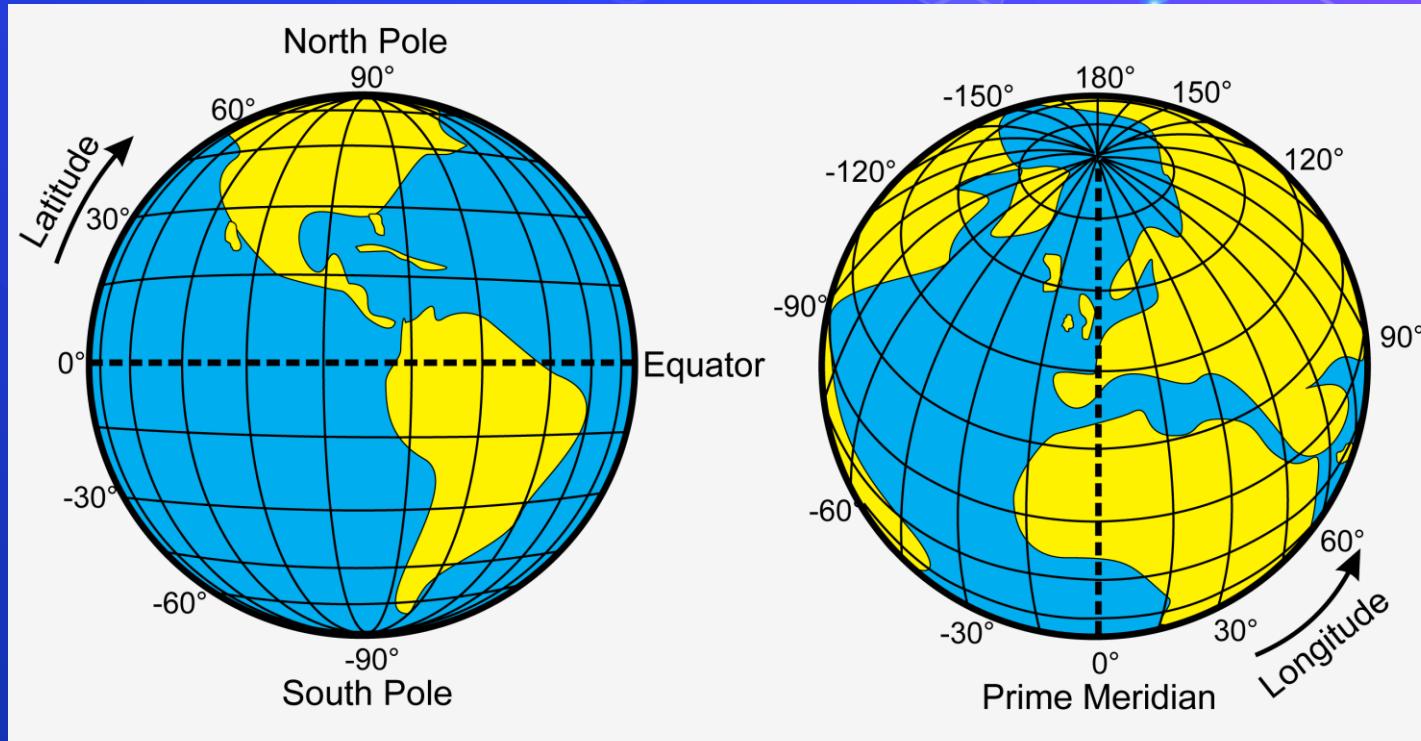
# Agenda

- hexagon icon What is Data
- hexagon icon Machine Learning
- hexagon icon Data Preprocessing
- hexagon icon **Feature Engineering and Extraction**
  - Domain knowledge features
  - Date and Time features
  - String operations
  - Web Data
  - **Geospatial features**
  - Work with Text
  - Work with Images
  - Work with Audio
  - Dimension Reduction with PCA

- hexagon icon Feature Transformations
  - Data Cleaning or Cleansing
  - Work with Missing data
  - Work with Categorical data
  - Detect and Handle Outliers
  - Split data to Train and Test Sets
  - Deal with Imbalanced classes
  - Feature Scaling
  - How to use Pipelines

- hexagon icon Feature Selection

# Geospatial features

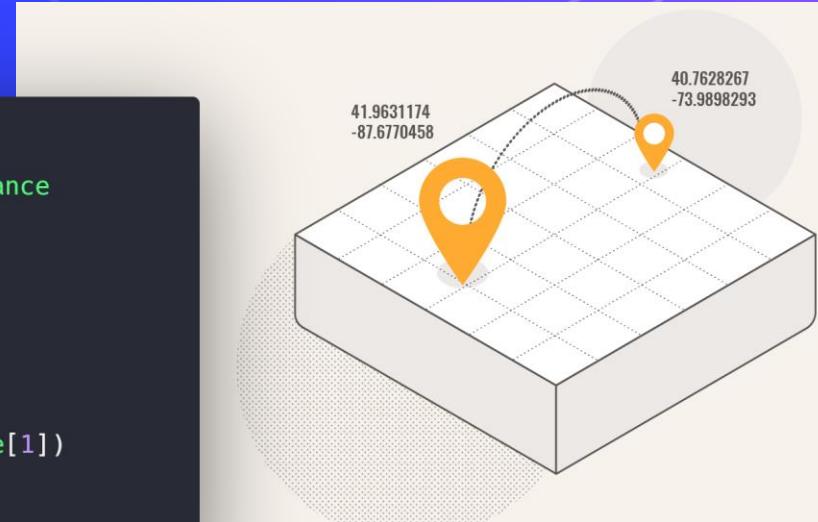


# Geospatial features

Measure distance with `haversine_distance`

<https://www.movable-type.co.uk/scripts/latlong.html>

```
1 from datasist.feature_engineering import haversine_distance
2
3
4 # lat, long
5 my_home = [30.109919, 31.308797]
6 cafe = [30.120982, 31.322026]
7
8
9 haversine_distance(my_home[0], my_home[1], cafe[0], cafe[1])
10 """
11 1.769848
12 """
```



# Agenda

- hexagon icon What is Data
- hexagon icon Machine Learning
- hexagon icon Data Preprocessing
- hexagon icon **Feature Engineering and Extraction**
  - Domain knowledge features
  - Date and Time features
  - String operations
  - Web Data
  - Geospatial features
  - Work with Text
  - Work with Images
  - Work with Audio
  - Dimension Reduction with PCA

- hexagon icon Feature Transformations
  - Data Cleaning or Cleansing
  - Work with Missing data
  - Work with Categorical data
  - Detect and Handle Outliers
  - Split data to Train and Test Sets
  - Deal with Imbalanced classes
  - Feature Scaling
  - How to use Pipelines
- hexagon icon Feature Selection

# Work with Text

## BAG OF WORDS

Converts text to a matrix where every row is an observation and every feature is a unique word. The value of each element in the matrix is either a binary indicator marking the presence of that word or an integer of the number of times that word appears.

ChrisAlbon



```
1 from sklearn.feature_extraction.text import CountVectorizer  
2  
3 vectorizer = CountVectorizer(stop_words='english')  
4 txt_feats = vectorizer.fit_transform(text)  
5 txt_feats.to_array()
```

# Work with Text

## TF-IDF

TF-IDF is a measure of originality of a word by comparing the number of times a word appears in a doc with the number of docs the word appears in.

$$\text{TF-IDF} = \text{TF}(t, d) \times \text{IDF}(t)$$

Term frequency  
Number of times term  $t$  appears in a doc,  $d$

$$\log \frac{1 + n}{1 + df(d, t)}$$

Document frequency of the term  $t$

```
1 from sklearn.feature_extraction.text import TfidfVectorizer
2
3 vectorizer = TfidfVectorizer(stop_words='english')
4 txt_feats = vectorizer.fit_transform(text)
5 txt_feats.toarray()
```

# Agenda

- hexagon icon What is Data
- hexagon icon Machine Learning
- hexagon icon Data Preprocessing
- hexagon icon **Feature Engineering and Extraction**
  - Domain knowledge features
  - Date and Time features
  - String operations
  - Web Data
  - Geospatial features
  - Work with Text
  - **Work with Images**
  - Work with Audio
  - Dimension Reduction with PCA

- hexagon icon Feature Transformations
  - Data Cleaning or Cleansing
  - Work with Missing data
  - Work with Categorical data
  - Detect and Handle Outliers
  - Split data to Train and Test Sets
  - Deal with Imbalanced classes
  - Feature Scaling
  - How to use Pipelines
- hexagon icon Feature Selection

# Work with Images

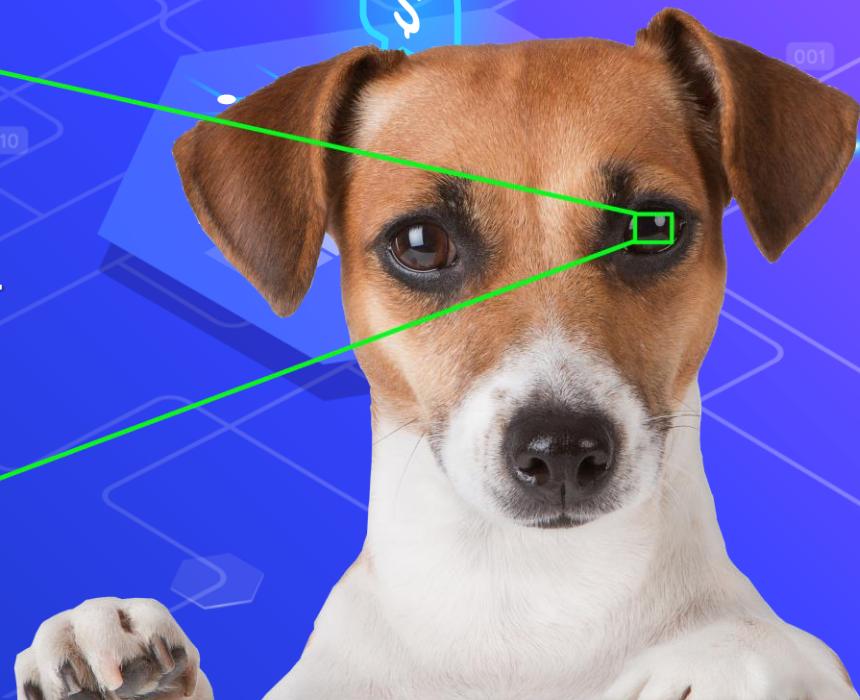
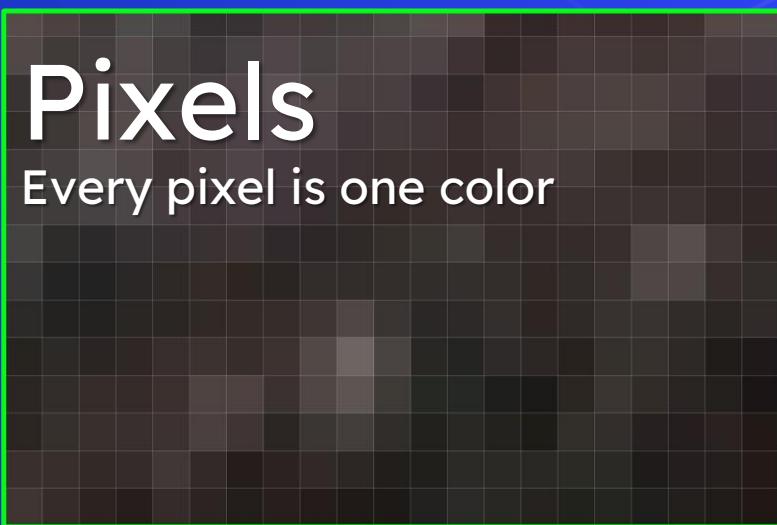


What We See

08 02 22 97 38 15 00 40 00 75 04 05 07 78 52 12 50 77 91 08 08 02 22 97  
49 49 99 40 17 81 18 57 60 87 17 40 98 43 69 48 04 56 62 00 49 49 99 40  
81 49 31 73 55 79 14 29 93 71 40 67 53 88 30 03 49 13 36 65 81 49 31 79  
52 70 95 23 04 60 11 42 69 24 68 56 01 32 56 71 37 02 36 91 52 70 95 23  
22 31 16 71 51 67 63 89 41 92 36 54 22 40 40 28 66 33 13 80 22 31 16 71  
24 47 32 60 99 03 45 02 44 75 33 53 78 36 84 20 35 17 12 50 24 47 32 60  
32 98 81 28 64 23 67 10 26 38 40 67 59 54 70 66 18 38 64 70 32 98 81 28  
67 26 20 68 02 62 12 20 95 63 94 39 43 08 40 91 66 49 94 21 67 26 20 68  
24 55 58 05 66 73 99 26 97 17 78 78 96 83 14 88 34 89 63 72 24 55 58 05  
21 36 23 09 75 00 76 44 20 45 35 14 00 61 33 97 34 31 33 95 21 36 23 09  
78 17 53 28 22 75 31 67 15 94 03 80 04 62 16 14 09 53 56 92 78 17 53 28  
16 39 05 42 96 35 31 47 55 58 88 24 00 17 54 24 36 29 85 57 16 39 05 42  
86 56 00 45 35 71 89 07 05 44 46 37 44 60 21 58 51 54 17 55 86 56 00 48  
19 80 81 68 05 94 47 69 25 73 92 13 86 52 17 77 04 89 55 40 19 80 81 68  
04 52 08 83 97 35 99 16 07 97 57 32 16 26 26 79 33 27 98 66 04 52 08 83  
88 36 68 87 57 62 20 72 03 46 33 67 46 55 12 32 63 93 33 69 88 36 68 87  
04 42 16 73 38 25 39 11 24 94 72 18 08 46 29 32 40 62 76 36 04 42 16 73  
20 69 36 41 72 30 23 88 34 62 99 69 82 67 59 85 74 04 36 16 20 69 36 41  
20 73 35 29 78 31 90 01 74 31 49 71 48 86 81 16 23 57 05 54 20 73 35 29  
01 70 54 71 83 51 54 69 16 92 33 48 61 43 52 01 89 19 67 48 01 70 54 71

What Computers See

# Work with Images



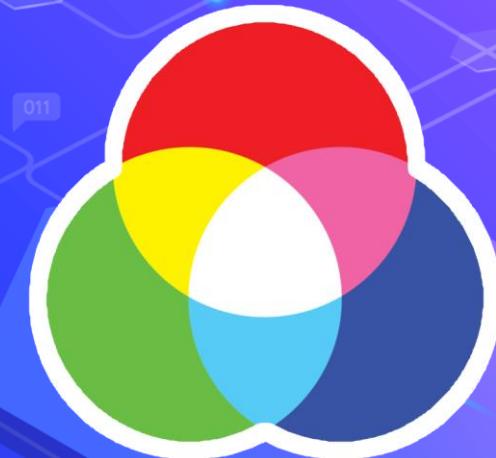
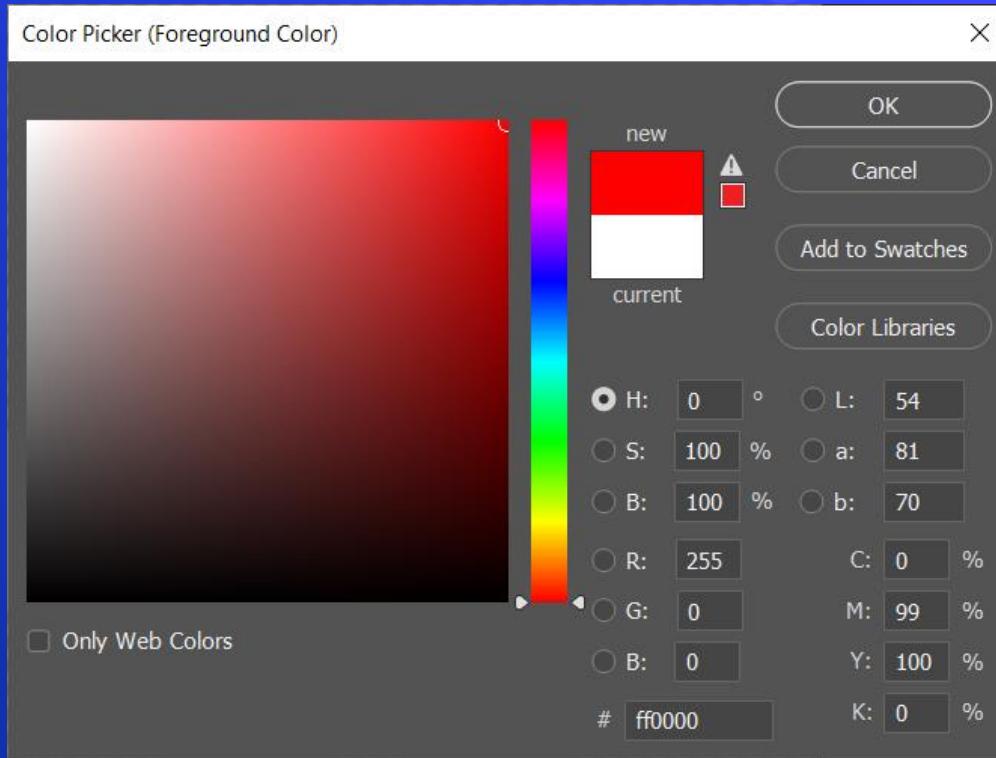
## Work with Images

# RGB Digital Images

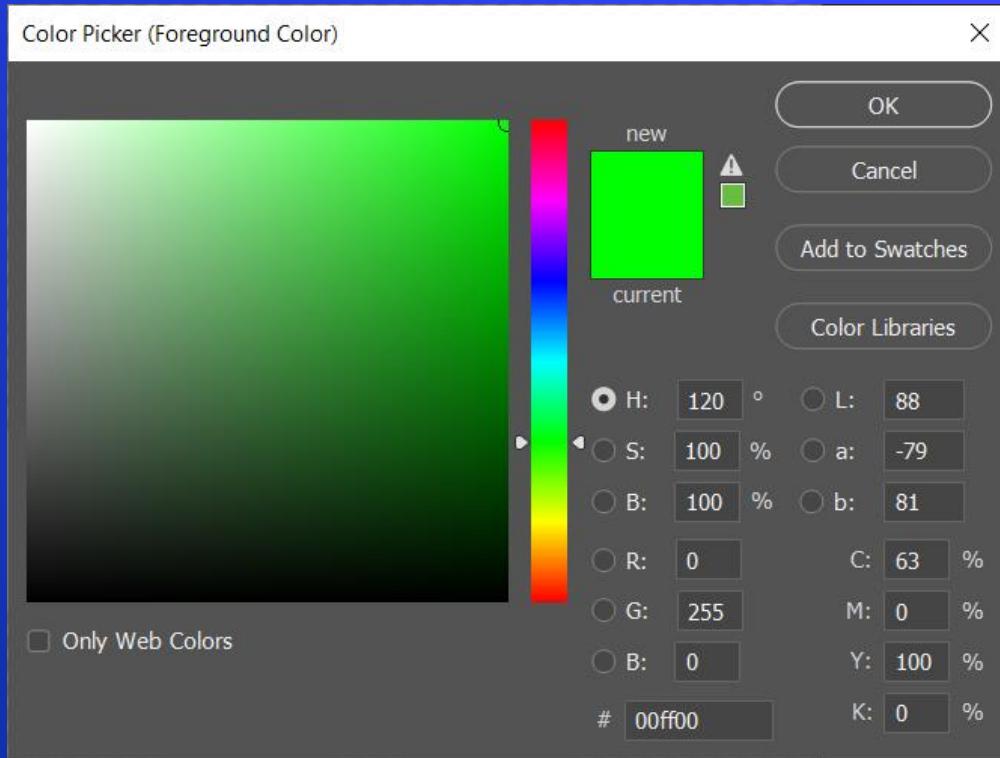
We can generate any color by mixing (red, green, blue) colors with different ratios.



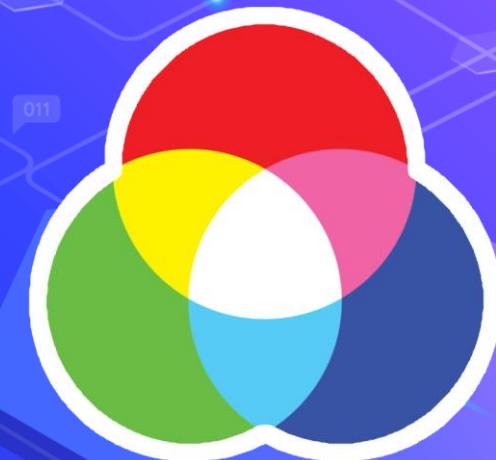
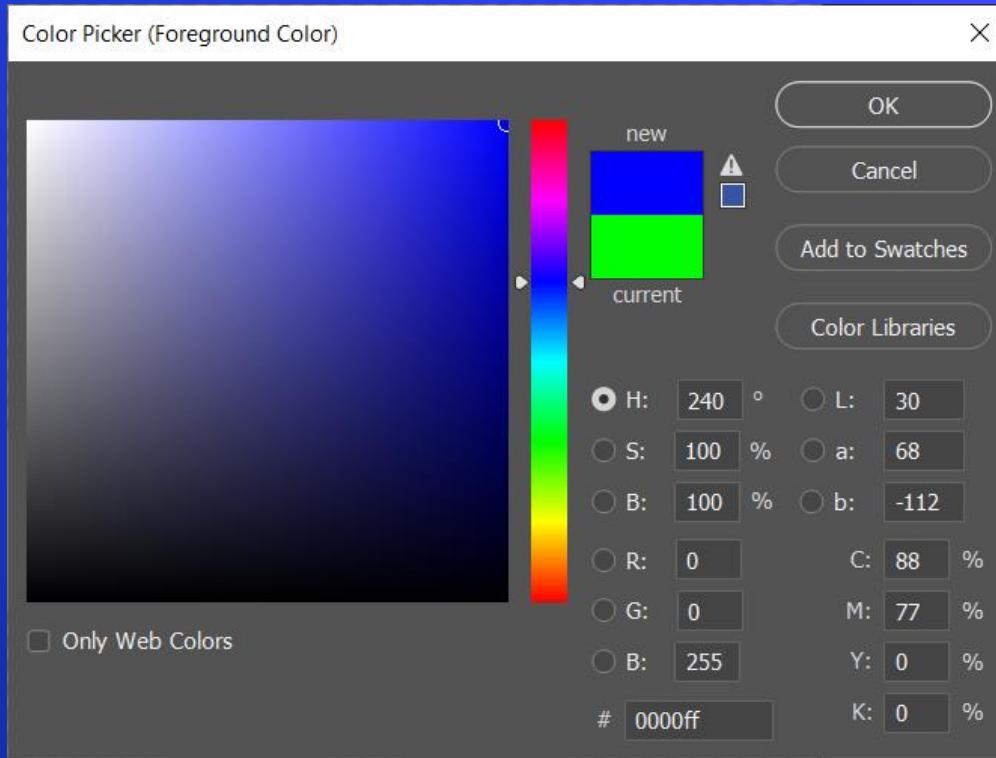
# Work with Images (Red)



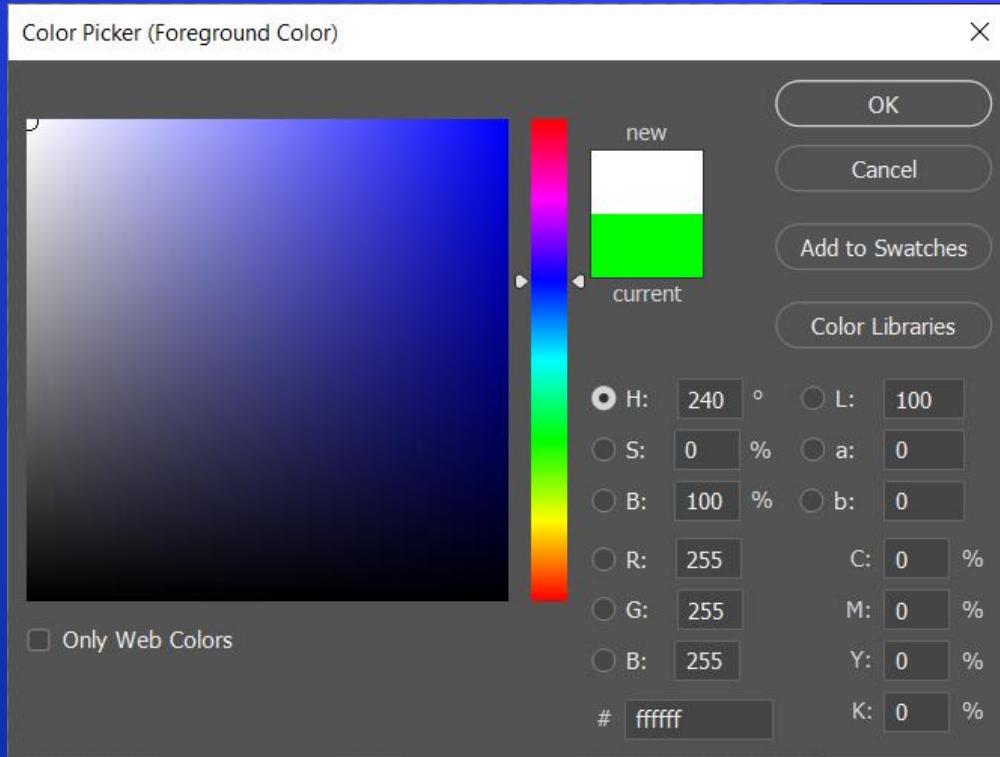
# Work with Images (Green)



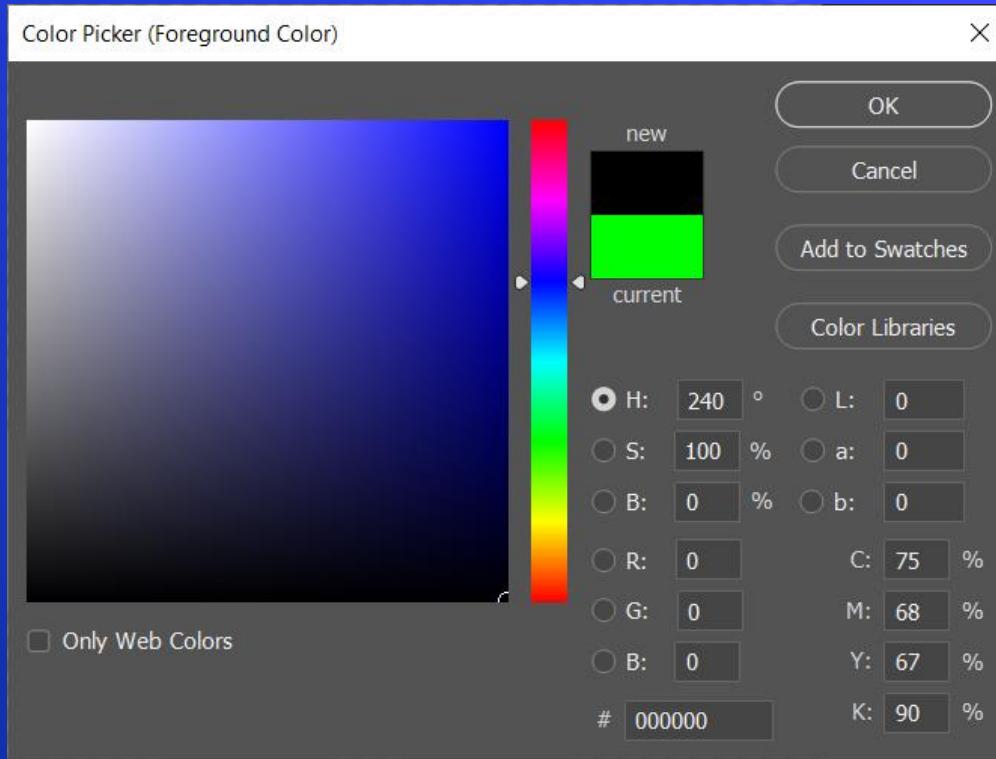
# Work with Images (Blue)



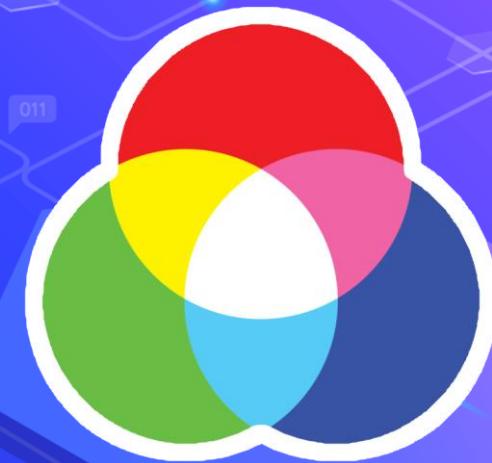
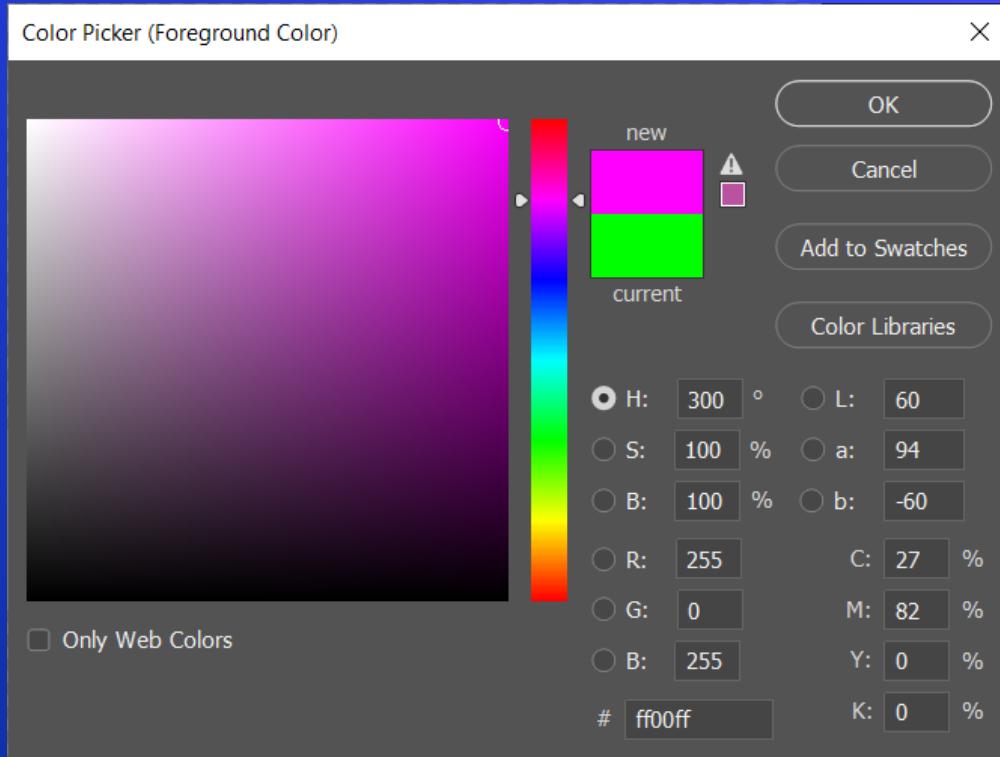
# Work with Images (White)



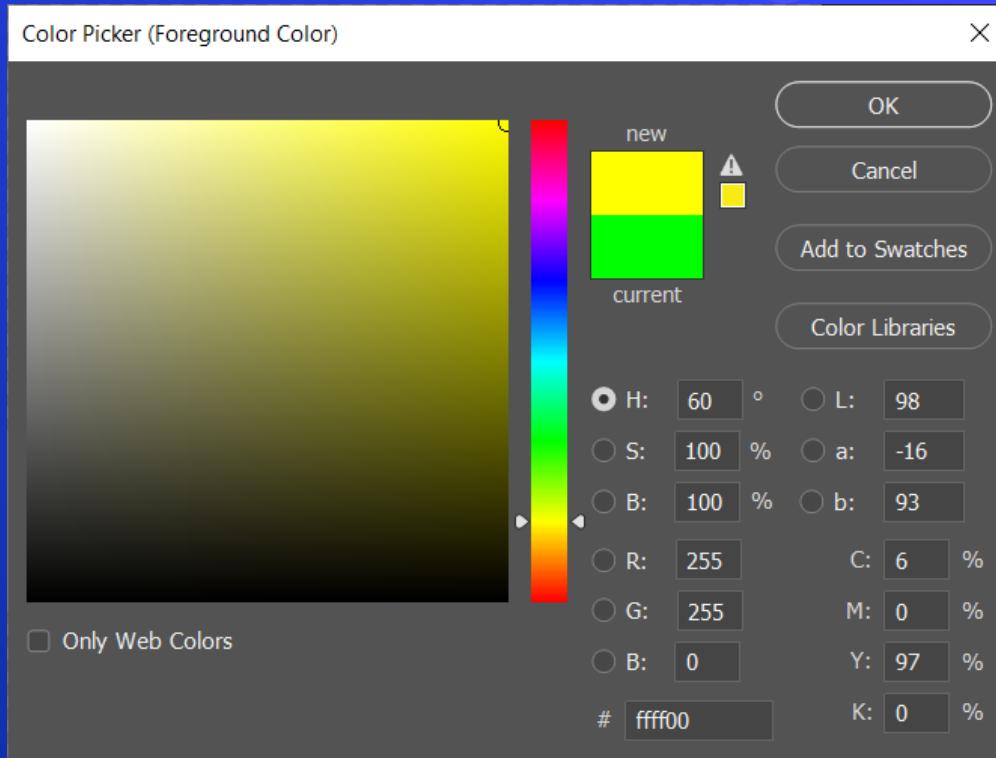
# Work with Images (Black)



# Work with Images (Pink)



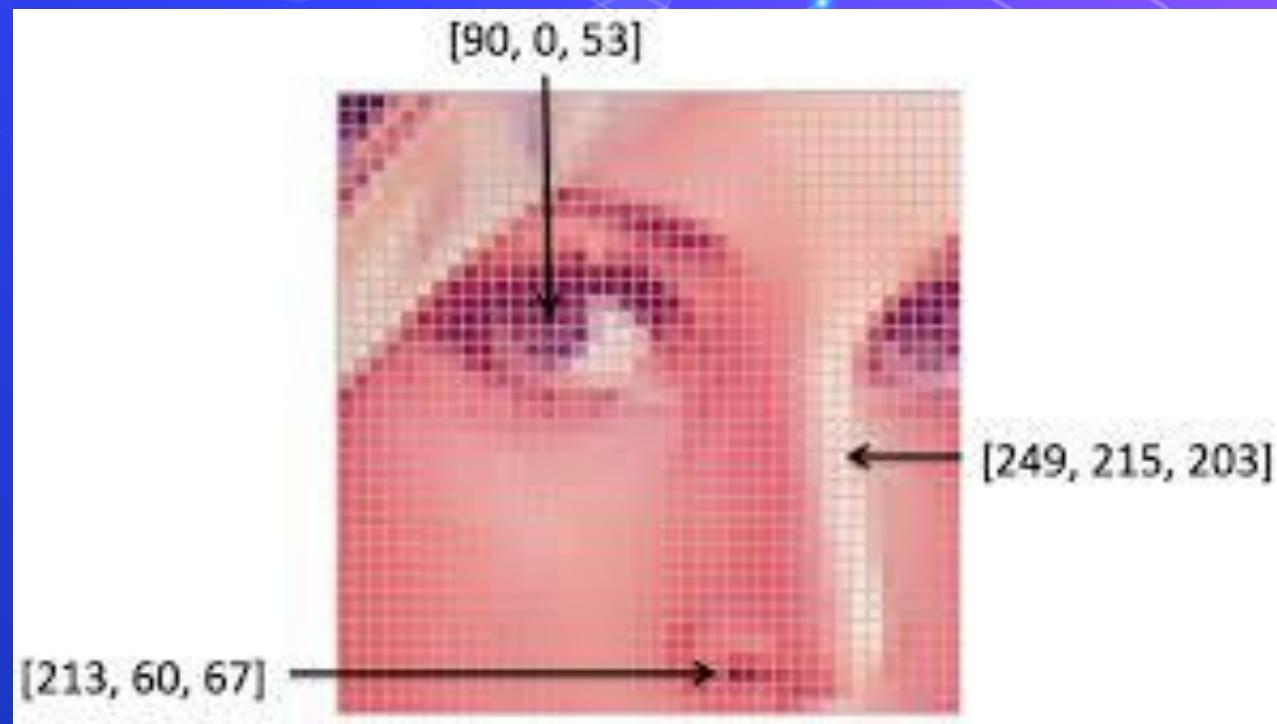
# Work with Images (Yellow)



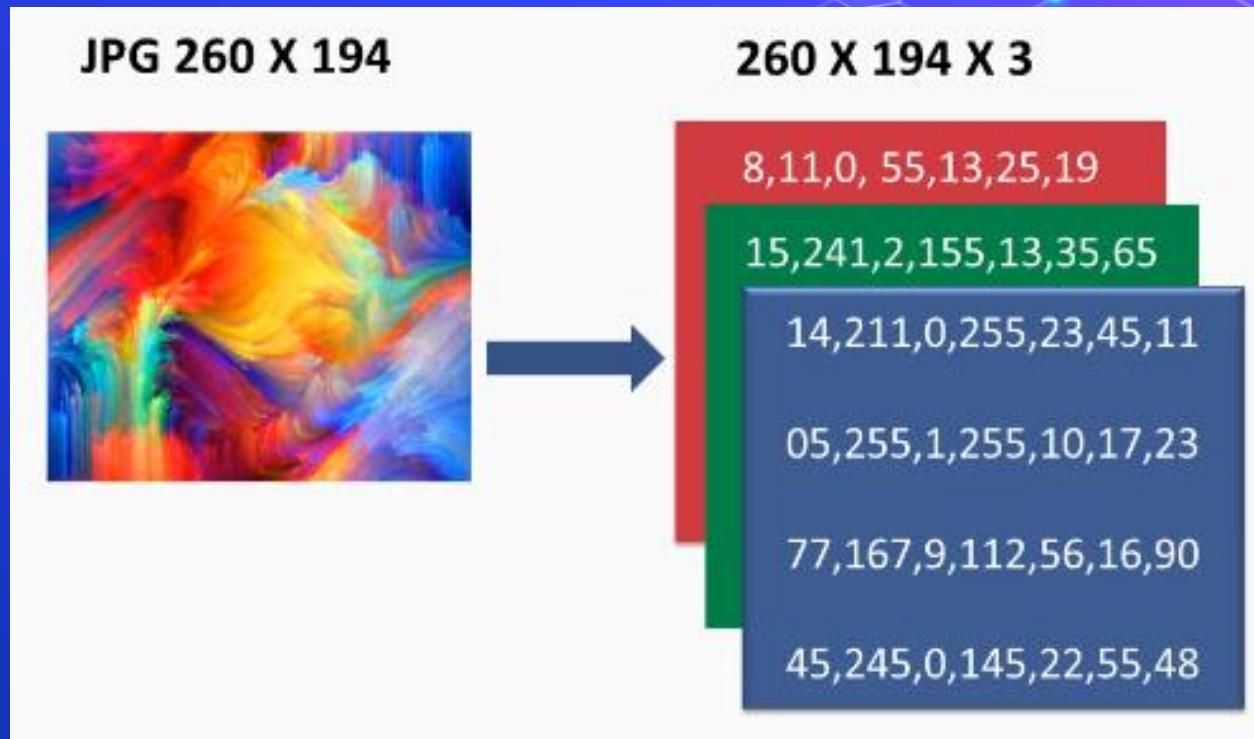
# Work with Images

Resolution: 100x100

Width : 100 pixels  
Height: 100 pixels



# Work with Images

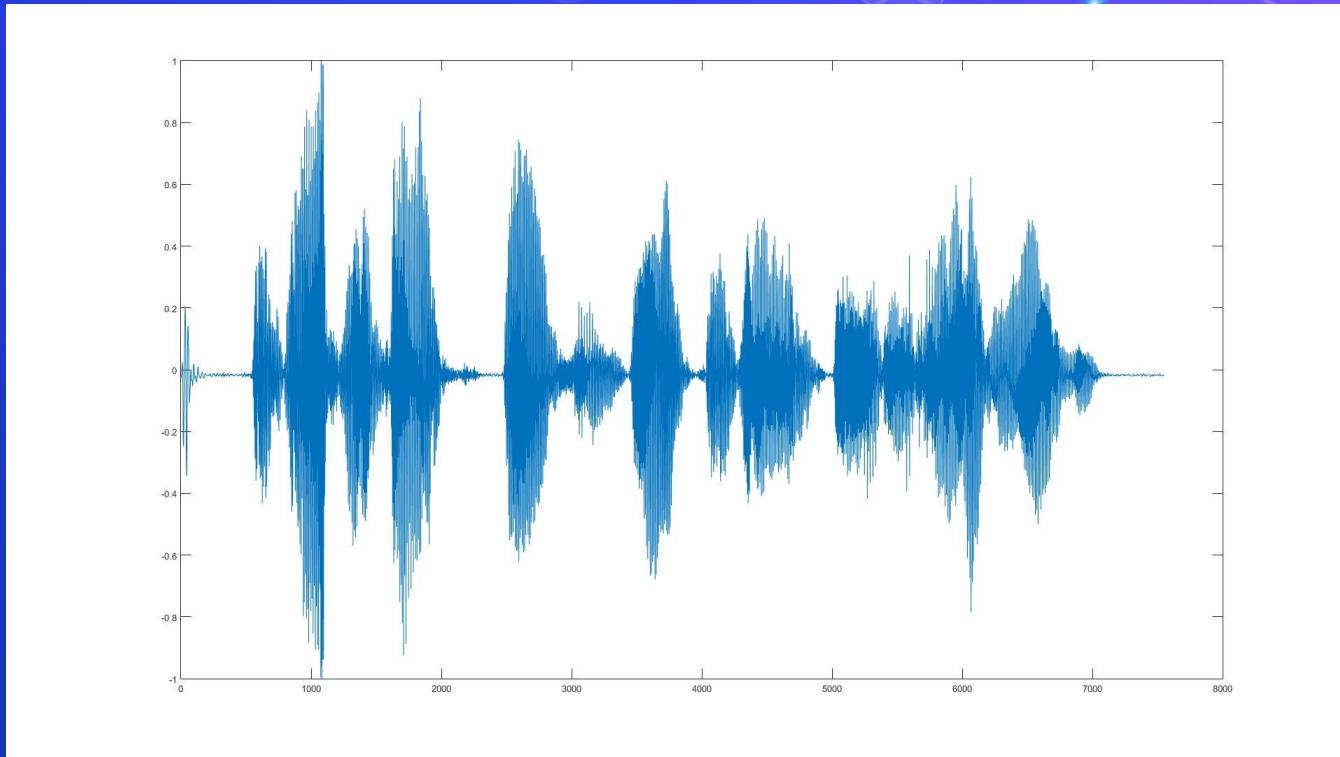


# Agenda

- hexagon icon What is Data
- hexagon icon Machine Learning
- hexagon icon Data Preprocessing
- hexagon icon **Feature Engineering and Extraction**
  - Domain knowledge features
  - Date and Time features
  - String operations
  - Web Data
  - Geospatial features
  - Work with Text
  - Work with Images
  - **Work with Audio**
  - Dimension Reduction with PCA

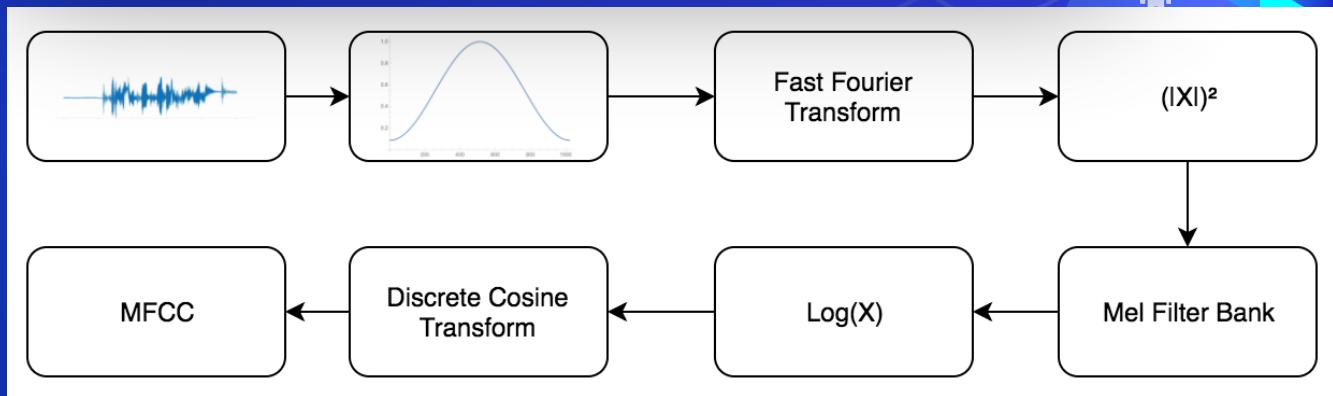
- hexagon icon Feature Transformations
  - Data Cleaning or Cleansing
  - Work with Missing data
  - Work with Categorical data
  - Detect and Handle Outliers
  - Split data to Train and Test Sets
  - Deal with Imbalanced classes
  - Feature Scaling
  - How to use Pipelines
- hexagon icon Feature Selection

# Work with Audio



# Work with Audio

```
1 import librosa  
2  
3 x, fs = librosa.load('sound.wav')  
4 mfccs = librosa.feature.mfcc(x, sr=fs)
```



[https://en.wikipedia.org/wiki/Mel-frequency\\_cepstrum](https://en.wikipedia.org/wiki/Mel-frequency_cepstrum)

# Agenda

- hexagon icon What is Data
- hexagon icon Machine Learning
- hexagon icon Data Preprocessing
- hexagon icon **Feature Engineering and Extraction**
  - Domain knowledge features
  - Date and Time features
  - String operations
  - Web Data
  - Geospatial features
  - Work with Text
  - Work with Images
  - Work with Audio
  - Dimension Reduction with PCA

- hexagon icon Feature Transformations
  - Data Cleaning or Cleansing
  - Work with Missing data
  - Work with Categorical data
  - Detect and Handle Outliers
  - Split data to Train and Test Sets
  - Deal with Imbalanced classes
  - Feature Scaling
  - How to use Pipelines
- hexagon icon Feature Selection

# Dimension Reduction with PCA

```
● ● ●  
1 from sklearn.decomposition import PCA  
2  
3 X = # feature vector  
4  
5 pca = PCA(0.9)  
6 X = pca.fit_transform(X)  
7 pca.explained_variance_ratio_
```

PRINCIPAL COMPONENT ANALYSIS

PCA projects the features onto the principal components. The motivation is to reduce the features dimensionality while only losing a small amount of information.

First principal component

Second principal component

[https://en.wikipedia.org/wiki/Principal\\_component\\_analysis](https://en.wikipedia.org/wiki/Principal_component_analysis)

# To be continued



# Questions ?!



# Thanks!

>\_ Live long and prosper

