# To explore the data visualization and data preprocesing

# Features & Its Types

 A **feature** is an individual measurable property or characteristic of a phenomenon being observed. Choosing informative, discriminating and independent **features** is a crucial step for effective algorithms in pattern recognition, classification and regression.

There are two types of features:

**Continuous Features**

A measurable difference exists between the values continuous features take on.Continuous variables are variables that can have an infinite number of possible values, as opposed to discrete variables which can only have a specified range of values.  Also continuous features are usually a subset of all real numbers. Some example features are:

* Time
* Distance
* Cost
* Temperature

**Categorical Features**

With categorical features, there is a specified number of discrete, possible feature values. These values may or may not have an ordering to them. If they do have a natural ordering, they are called ordinal categorical features. Otherwise if there is no intrinsic ordering, they are called nominal categorical features.

**Nominal**

* Car Models
* Colors
* TV Shows

**Ordinal**

* High-Medium-Low
* 1-10 Years Old, 11-20 Years Old, 30-40 Years Old
* Happy, Neutral, Sad

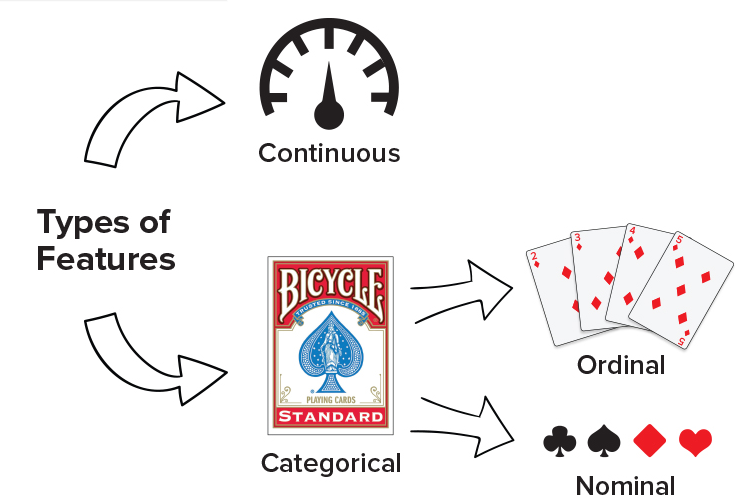
s

Figure 3.1: Types of features

## Visualizations

One of the most rewarding and useful things you can do to understand your data is to visualize it in a pictorial format. Visualizing your data allows you to interact with it, analyze it in a straightforward way, and identify new patterns, making your would-be complex data more accessible and understandable. The way our brains processes visuals like shapes, colors, and lengths makes looking at charts and graphs more intuitive for us than poring over spreadsheets.

### Matplotlib

MatPlotLib is a Python data visualization tool that supports 2D and 3D rendering, animation, UI design, event handling, and more. It only requires you pass in your data and some display parameters and then takes care of all of the rasterization implementation details. For the most part, you will be interacting with MatPlotLib's Pyplot functionality through a Pandas series or dataframe's .plot namespace. Pyplot is a collection of command-style methods that essentially make MatPlotLib's charting methods feel like MATLAB.

## Basic Plots

### Histograms

Histograms are graphical techniques which have been identified as being most helpful for troubleshooting issues. Histograms help you understand the distribution of a feature in your dataset. They accomplish this by simultaneously answering the questions where in your feature's domain your records are located at, and how many records exist there.

Let's go ahead and explore a little bit about how to use histograms and what they can actually do. You've probably seen wheat before, nothing new there, however, it turns out that there's quite a few different varieties of it. A group titled the Polish Academy of Science, particularly their Agrophysics Institute, what they do is they created a dataset that has a bunch of different types of wheat in them. And they x-rayed the wheat, the different specimens of wheat and then they featured the results. So some of the metrics that they curated, include the groove length of the wheat which is this thing over here. They also got the actual kernel length and the kernel width and some other features about the wheat. In addition to that, they also added in some engineered or calculated features such as an asymmetry constant.

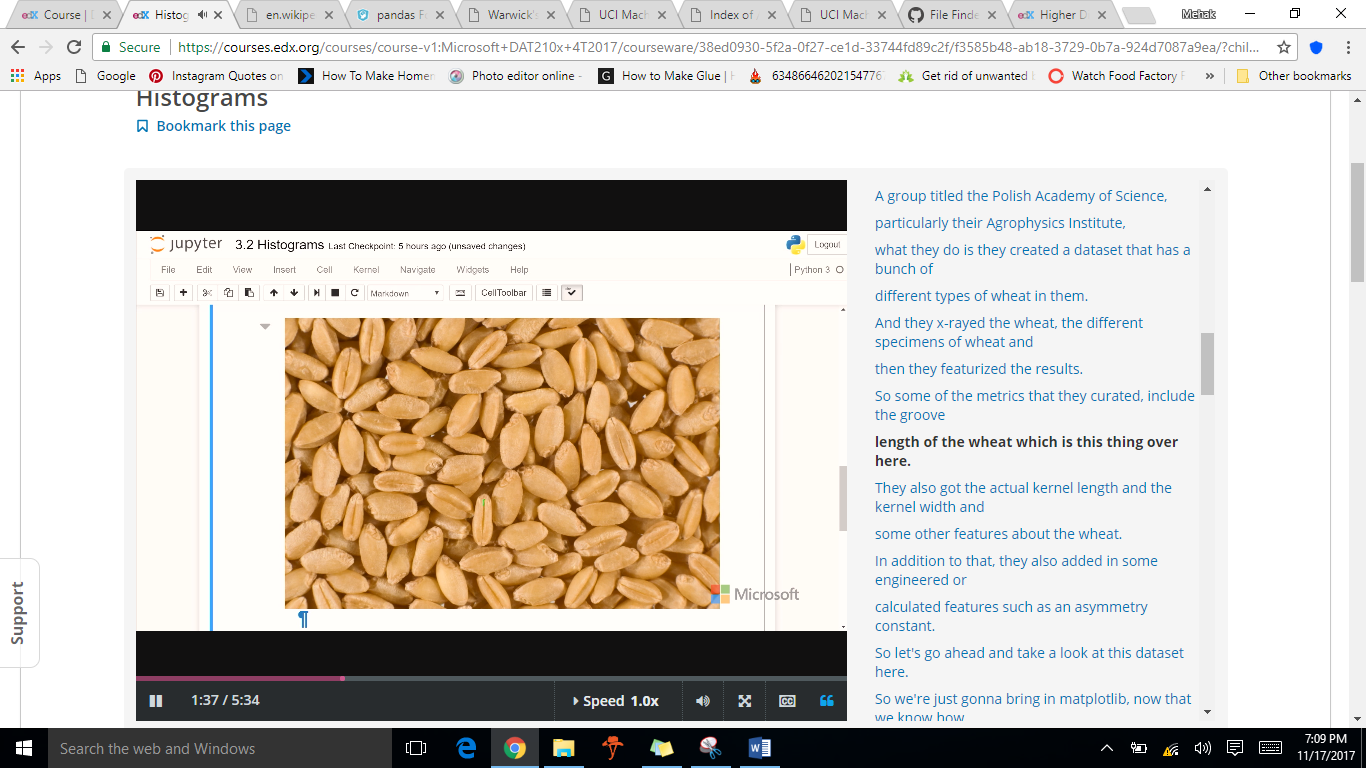


Figure 3.2: Wheat Kernel

DATA SET DESCRIPTION

The examined group comprised kernels belonging to three different varieties of wheat: Kama, Rosa and Canadian, 70 elements each, randomly selected for   
the experiment. High quality visualization of the internal kernel structure was detected using a soft X-ray technique. It is non-destructive and considerably cheaper than other more sophisticated imaging techniques like scanning microscopy or laser technology. The images were recorded on 13x18 cm X-ray KODAK plates. Studies were conducted using combine harvested wheat grain originating from experimental fields, explored at the Institute of Agrophysics of the Polish Academy of Sciences in Lublin.



|  |
| --- |
| Example 3.1: |
| import pandas as pd  import matplotlib  import matplotlib.pyplot as plt  matplotlib.style.use('ggplot') # Look Pretty, ggplot is Python implementation of the grammar of graphics.  # If the above line throws an error, use plt.style.use('ggplot') instead    df = pd.read\_csv("C:/Users/Mehak/Desktop/DataScience/wheat.csv", index\_col=0)  #Prints columns in present in your csv file  print(df.columns)  #Creates a squence named as my\_series\_series using asymetry column  my\_series = df.asymmetry  #Creates a dataframe named as my\_dataframe using provided columns  my\_dataframe = df[['wheat\_type', 'length', 'asymmetry']]    #Histogram creation of sequence  my\_series.plot.hist(alpha=0.5)  plt.show()  #Histogram creation of dataframe  my\_dataframe.plot.hist(alpha=0.5)  plt.show()  #Histogram creation based on particular condition  df[df.wheat\_type==1].asymmetry.plot.hist(alpha=0.4)#Kama  df[df.wheat\_type==2].asymmetry.plot.hist(alpha=0.4)#Rosa  df[df.wheat\_type==3].asymmetry.plot.hist(alpha=0.4)#Canadian  plt.show() |
| Output: |
| Figure 3.3: Histogram based on one feature  Chart, histogram  Description automatically generated  Figure 3.4: Histogram based on three features  Chart, histogram  Description automatically generated  Figure 3.5: Histogram based on conditions |

### 2D Scatter Plots

2D scatter plots are used to visually inspect if a correlation exist between the charted features. Both axes of a 2D scatter plot represent a distinct, numeric feature. They don't have to be continuous, but they must at least be ordinal since each record in your dataset is being plotted as a point with its location along the axes corresponding to its feature values. Without ordering, the position of the plots would have no meaning.

It is possible that either a negative or positive correlation exist between the charted features, or alternatively, none at all. The correlation type can be assessed through the overall diagonal trending of the plotted points.

Positive and negative correlations may further display a linear or non-linear relationship. If a straight line can be drawn through your scatter plot and most of points seem to stick close to it, then it can be said with a certain level of confidence that there is a linear relationship between the plotted features. Similarly, if a curve can be drawn through the points, there is likely a non-linear relationship. If neither a curve nor line adequately seems to fit the overall shape of the plotted points, chances are there is neither a correlation nor relationship between the features, or at least not enough information at present to determine.

Dataset for the next example is taken from UCI Machine Learning Repository.



|  |
| --- |
| Example 3.2: |
| import pandas as pd  import matplotlib  import matplotlib.pyplot as plt    matplotlib.style.use('ggplot') # Look Pretty  # If the above line throws an error, use plt.style.use('ggplot') instead    #Creation of dataframe from csv file  df = pd.read\_csv("C:/Users/Mehak/Desktop/DataScience/Concrete\_Data.csv", index\_col=0)  #prints columns in file  print(df.columns)  #Rename Columns  df.columns=['Slag','Ash','Water','Superplasticizer','Coarse Aggregate','Fine Aggregate','Age','Strength']  #Print new column names  print(df.columns)  #Creates scatter plots based on different features  df.plot.scatter(x='Slag', y='Strength')  plt.show()  df.plot.scatter(x='Water', y='Strength')  plt.show()  df.plot.scatter(x='Ash', y='Strength')  plt.show() |
| Output: |
| Figure 3.6: 2D Scatter plot based on slag & strength    Figure 3.6: 2D Scatter plot based on water & strength    Figure 3.6: 2D Scatter plot based on ash & strength |

### 3D Scatter Plots

There surely is a way to visualize the relationship between three variables simultaneously. That way is through 3D scatter plots. Unfortunately, the Pyplot member of Pandas data frames don't natively support the ability to generate 3D plots, so for the sake of your visualization repertoire, you're going to learn how to make them directly with MatPlotLib.

|  |
| --- |
| Example 3.3: |
| import pandas as pd  import matplotlib  import matplotlib.pyplot as plt  from mpl\_toolkits.mplot3d import Axes3D #for creation of 3d plots    matplotlib.style.use('ggplot') # Look Pretty  # If the above line throws an error, use plt.style.use('ggplot') instead    #Creation of dataframe from csv file  df = pd.read\_csv("C:/Users/Mehak/Desktop/DataScience/Concrete\_Data.csv", index\_col=0)  #prints columns in file  print(df.columns)  #Rename Columns  df.columns=['Slag','Ash','Water','Superplasticizer','Coarse Aggregate','Fine Aggregate','Age','Strength']  #Print new column names  print(df.columns)  #creates figure  fig = plt.figure()  ax = fig.add\_subplot(111, projection='3d')#"1x1 grid, first subplot"  #OR USE ax = fig.gca(projection='3d'),gca(Get the current axes, creating one if necessary)  ax.set\_xlabel('Slag')  ax.set\_ylabel('Water')  ax.set\_zlabel('Strength')    ax.scatter(df.Slag, df.Water, df.Strength, c='b', marker='.')  plt.show() |
| Output: |
| Figure 3.7: 3D Scatter plot |

## Higher Dimensionality Visualizations