INDIAN INSTITUTE OF TECHNOLOGY MADRAS • सिद्धिर्भवति कर्मजा •

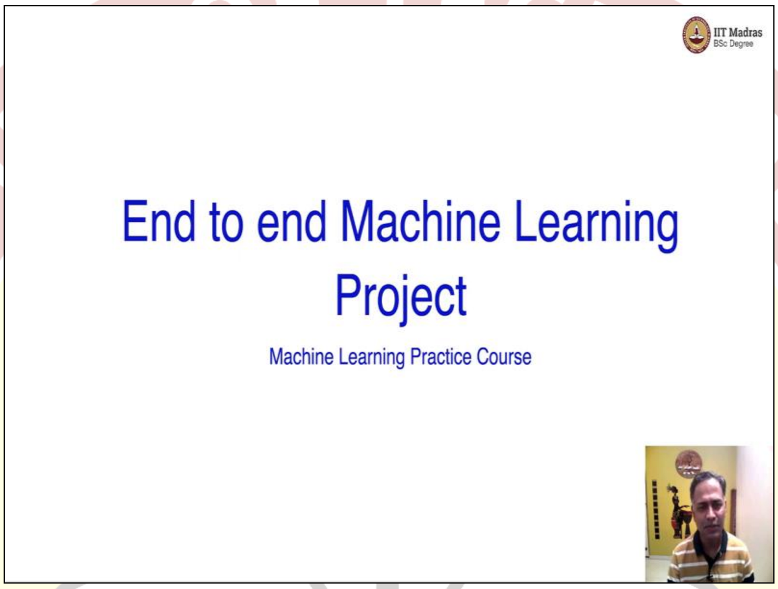
IIT Madras

ONLINE DEGREE

Machine Learning Practice
Online Degree Programme
B. Sc in Programming and Data Science
Diploma Level
Dr. Ashish Tendulkar
Indian Institute of Technology – Madras

Introduction Looking at the Big Picture

(Refer Slide Time: 00:10)



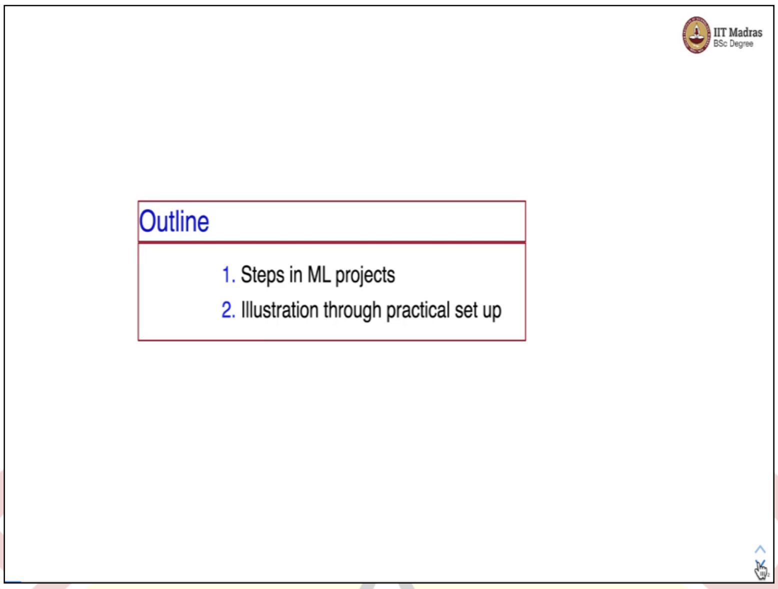
End to end Machine Learning Project

Machine Learning Practice Course

Namaste welcome to the machine learning practice course. My name is Ashish Tendulkar and I will be your instructor for this course. This course is running in parallel with the machine learning techniques course while the technique course focuses on the theoretical aspect of machine learning. The practice course will focus on practical aspects of machine learning. Specifically, we will be using a sklearn python library for implementing machine learning algorithms that we study in the machine learning techniques course.

This course is designed to make you industry-ready. So, at the end of this course, you should be able to design machine learning algorithms for practical situations. So, the first topic of this course is end to end Machine learning project.

(Refer Slide Time: 01:05)



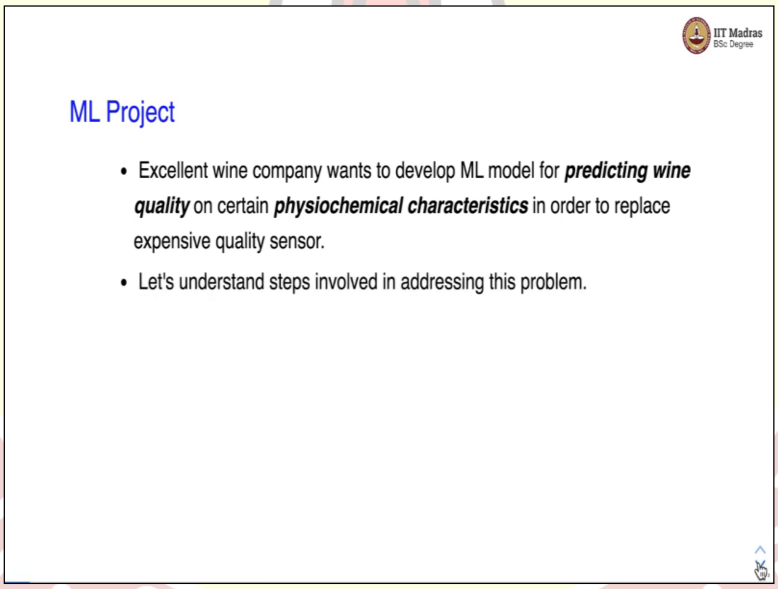
IIT Madras
BSc Degree

Outline

1. Steps in ML projects
2. Illustration through practical set up

In this topic, we will study steps in a typical machine learning project. We illustrate these steps through a practical setup.

(Refer Slide Time: 01:14)



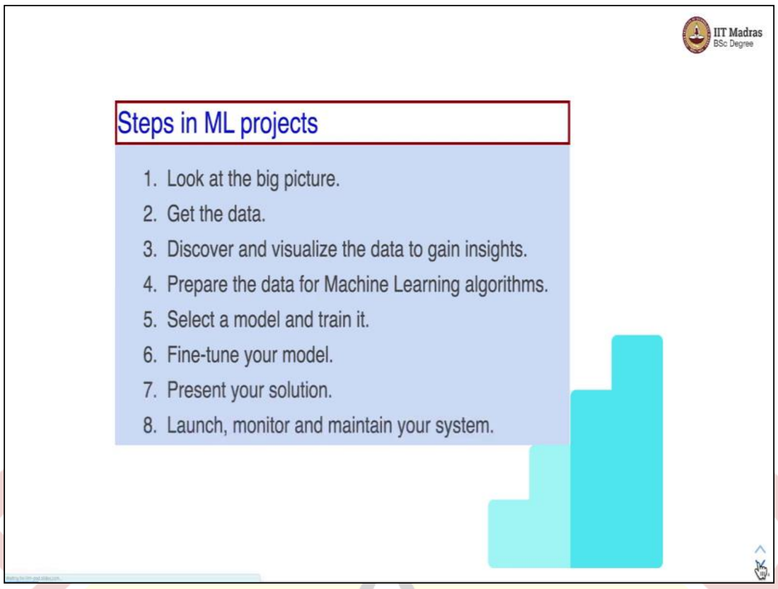
IIT Madras
BSc Degree

ML Project

- Excellent wine company wants to develop ML model for ***predicting wine quality*** on certain ***physiochemical characteristics*** in order to replace expensive quality sensor.
- Let's understand steps involved in addressing this problem.

So, as a practical setup consider that there is an excellent wine company that wants to develop a machine learning model for predicting wine quality based on certain physiochemical characteristics of the wine they want to use this machine learning model to replace the expensive quality sensor. Let us understand the steps involved in addressing this problem.

(Refer Slide Time: 01:42)

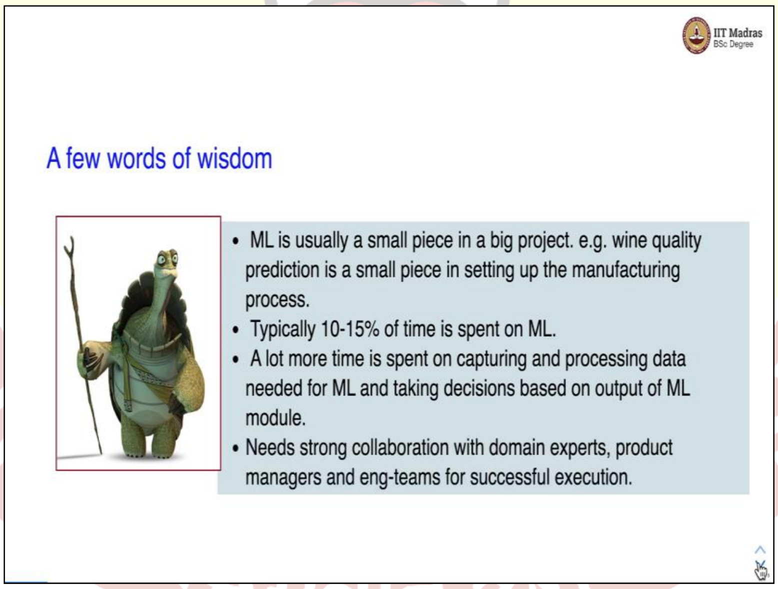


Steps in ML projects

1. Look at the big picture.
2. Get the data.
3. Discover and visualize the data to gain insights.
4. Prepare the data for Machine Learning algorithms.
5. Select a model and train it.
6. Fine-tune your model.
7. Present your solution.
8. Launch, monitor and maintain your system.

There are eight steps in a machine learning project. First, we look at the big picture. Then we get the data, discover and visualize the data to gain insights we prepare the data for machine learning algorithms, select a model, and train it. Then we fine-tune our model, we present our solution and finally, we launch monitor and maintain our system.

(Refer Slide Time: 02:12)



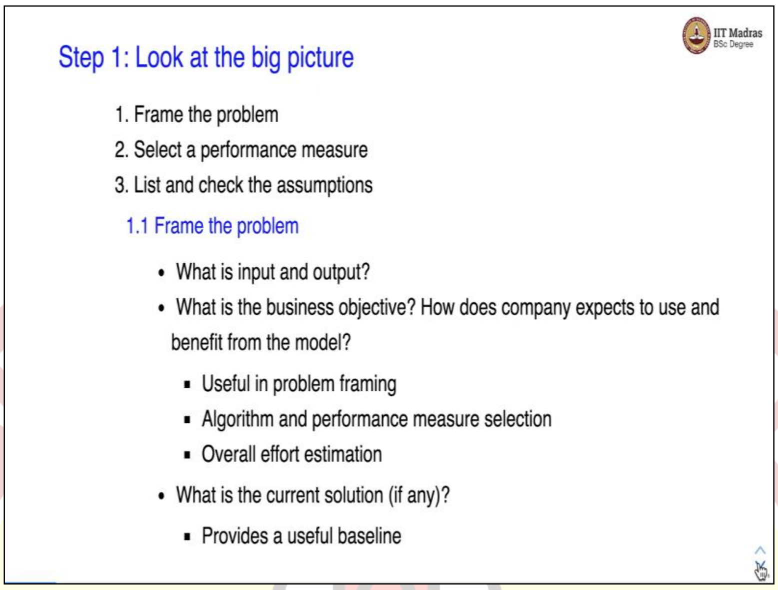
A few words of wisdom

- ML is usually a small piece in a big project. e.g. wine quality prediction is a small piece in setting up the manufacturing process.
- Typically 10-15% of time is spent on ML.
- A lot more time is spent on capturing and processing data needed for ML and taking decisions based on output of ML module.
- Needs strong collaboration with domain experts, product managers and eng-teams for successful execution.

So, generally, machine learning is a small piece in a big project. For example, wine quality prediction is a small piece in setting up the manufacturing process. We spend typically 10 to 15% of the time on machine learning setup. A lot more time is spent on capturing and processing data needed for machine learning and taking decisions based on the output of the machine learning module.

And we need strong collaboration with domain expert's product managers and engineering teams for the successful execution of machine learning projects.

(Refer Slide Time: 02:54)



Step 1: Look at the big picture

1. Frame the problem
2. Select a performance measure
3. List and check the assumptions

1.1 Frame the problem

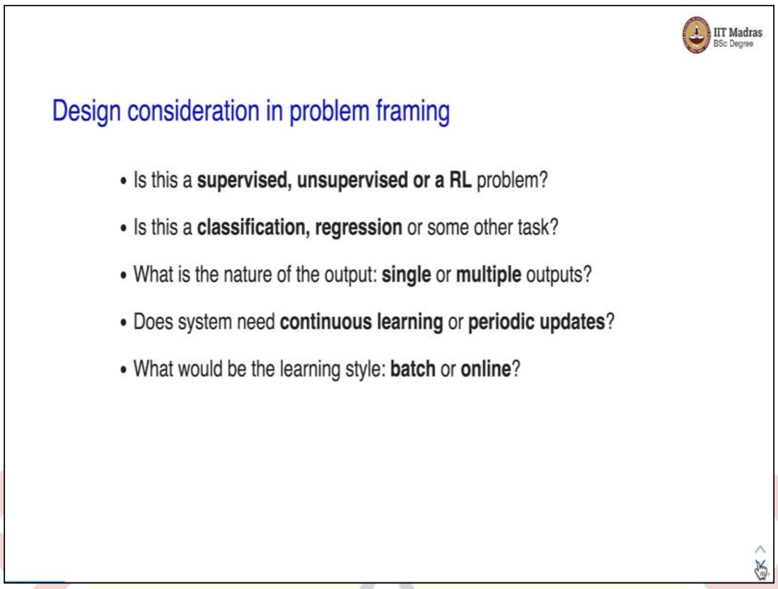
- What is input and output?
- What is the business objective? How does company expects to use and benefit from the model?
  - Useful in problem framing
  - Algorithm and performance measure selection
  - Overall effort estimation
- What is the current solution (if any)?
  - Provides a useful baseline

Let us look at the first step where we look at the big picture of the problem. Here the first and most important thing is to frame the problem properly. We select a performance measure then we list and check the assumptions that we have made while arriving at the solution. As part of framing the problem, we need to figure out what is the input and output.

What is the business objective and how does the company expects to use and benefit from the model? These measures or this kind of information are useful in problem framing. It is also useful in algorithm and performance major selection. And it also aids in overall effort estimation. It is also important to understand what is the current solution? Because the current solution provides a useful baseline sometimes the current solution could be to use human labor to figure out what is the quality of the wine?

Now we can figure out what is the accuracy or what is the accuracy of the current system that is deployed for the problem and we aspire to build a basic machine learning model that attains at least that level of accuracy as a baseline. And if we provide all the data that is available to this particular process we hope to build a machine learning model that is likely to beat this particular baseline.

(Refer Slide Time: 04:43)



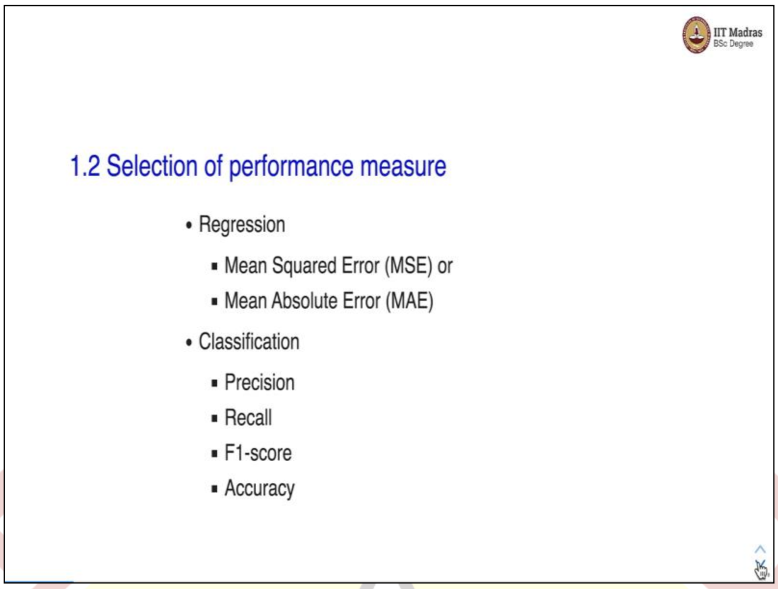
Design consideration in problem framing

- Is this a **supervised, unsupervised or a RL** problem?
- Is this a **classification, regression** or some other task?
- What is the nature of the output: **single** or **multiple** outputs?
- Does system need **continuous learning** or **periodic updates**?
- What would be the learning style: **batch** or **online**?

Let us look at some of the design considerations in problem framing. We need to figure out whether the problem is a supervised learning problem unsupervised learning problem or a reinforcement learning problem. You are aware of supervised and unsupervised learning problems from the machine learning techniques course. So, as part of a supervised learning task whether it is a classification or regression task or some other task. What is the nature of the output whether there is a single output or multiple outputs that are expected from the machine learning system?

Does the system need to learn continuously or if the periodic updates are sufficient? And what would be the learning style whether we will be learning in a batch or whether we will be learning in online fashion in online fashion as the data is arriving we need to we are supposed to learn the model we are supposed to update our model and use that updated model to make predictions. In the batch mode, we take a bunch of examples together in order to learn the model.

(Refer Slide Time: 05:57)

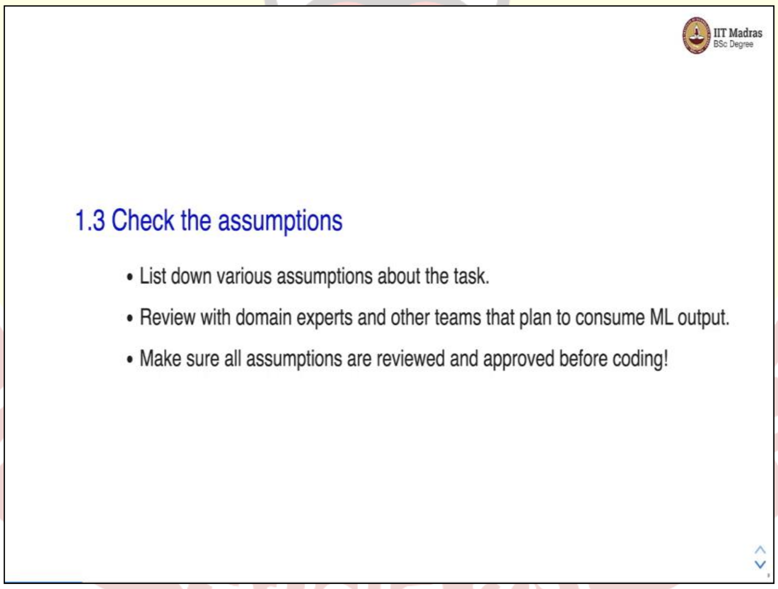


1.2 Selection of performance measure

- Regression
  - Mean Squared Error (MSE) or
  - Mean Absolute Error (MAE)
- Classification
  - Precision
  - Recall
  - F1-score
  - Accuracy

The next part of the process is to select the appropriate performance measure. For regression, we use mean squared error or mean absolute error as performance measures. For classification,n we have precision-recall F1-score and accuracy as performance measures.

(Refer Slide Time: 06:26)

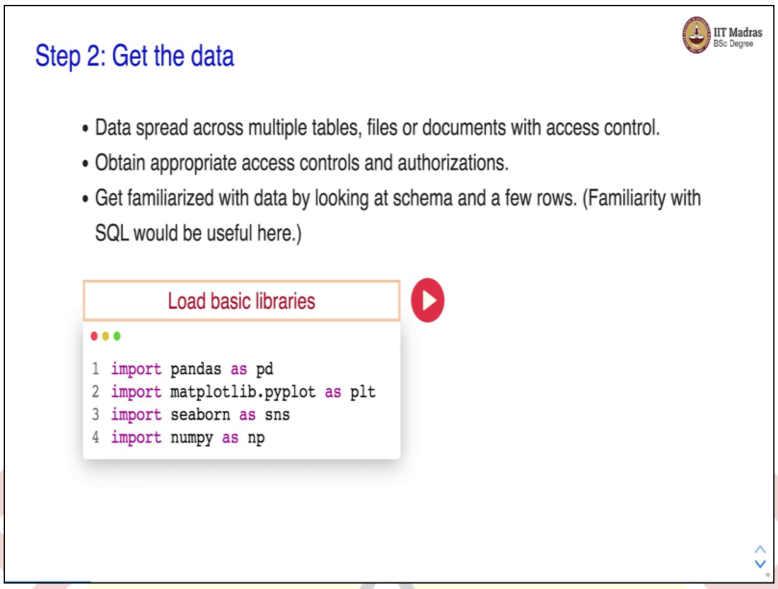


1.3 Check the assumptions

- List down various assumptions about the task.
- Review with domain experts and other teams that plan to consume ML output.
- Make sure all assumptions are reviewed and approved before coding!

The next step is to check the assumptions that we make for solving the problem. We need to list down various assumptions about the task we would like to review these assumptions with domain experts and other teams that plan to consume the output of the machine learning module. This helps us in making sure that all assumptions are reviewed and approved before coding.

(Refer Slide Time: 07:00)



Step 2: Get the data

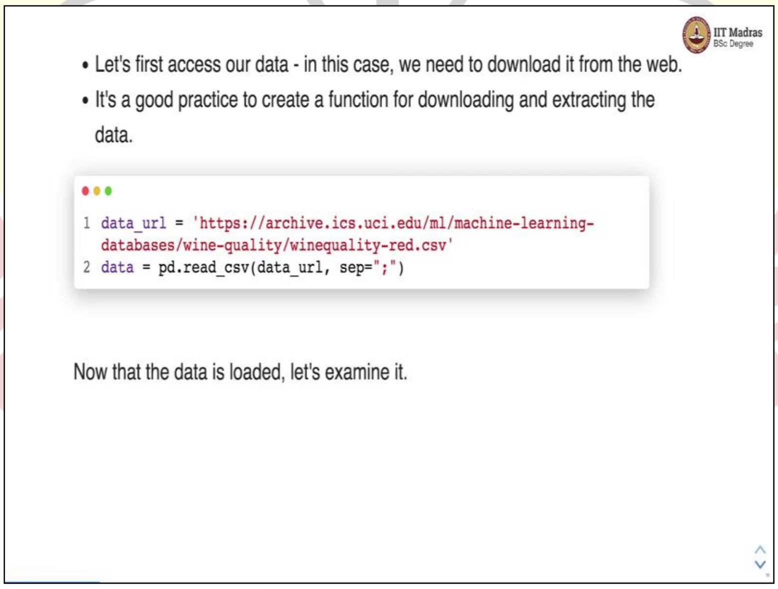
- Data spread across multiple tables, files or documents with access control.
- Obtain appropriate access controls and authorizations.
- Get familiarized with data by looking at schema and a few rows. (Familiarity with SQL would be useful here.)

Load basic libraries

```python
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import numpy as np
```

The second step of the process is to get the data. The data is usually spread across multiple tables files or documents with access control. We need to obtain appropriate access controls and authorizations to access this data. Then we generally get familiarized with data by looking at the schema and a few rows. In this step familiarity with SQL would be very useful. We first load the basic python libraries that are needed for our prototype solution.

(Refer Slide Time: 07:45)



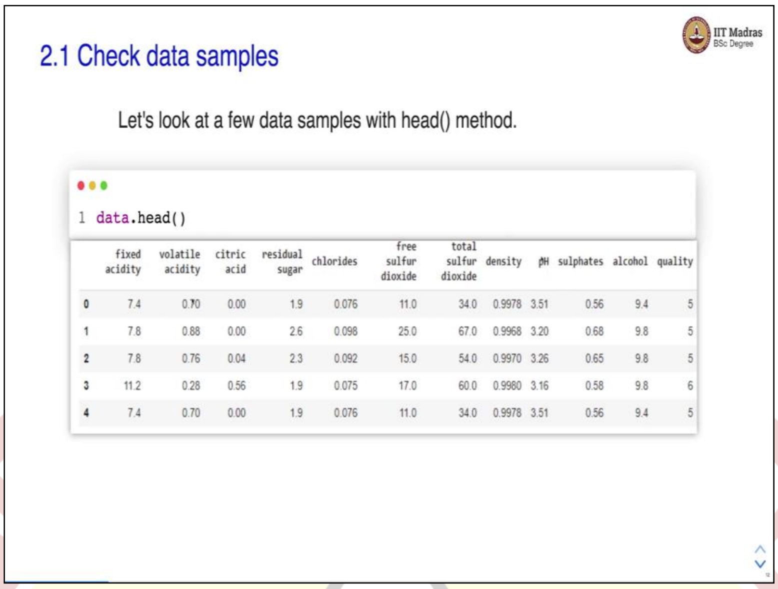
- Let's first access our data - in this case, we need to download it from the web.
- It's a good practice to create a function for downloading and extracting the data.

```python
data_url = 'https://archive.ics.uci.edu/ml/machine-learning-databases/wine-quality/winequality-red.csv'
data = pd.read_csv(data_url, sep=";")
```

Now that the data is loaded, let's examine it.

Let us first access our data in this case we need to download this data from the web. It is a good practice to create a function for downloading and extracting the data. Here we download the data from UCI machine learning repository and the data is in the form of a comma separated values or in a csv format. Now that the data is loaded we need to examine it.

(Refer Slide Time: 08:18)



IIT Madras
BSc Degree

2.1 Check data samples

Let's look at a few data samples with head() method.

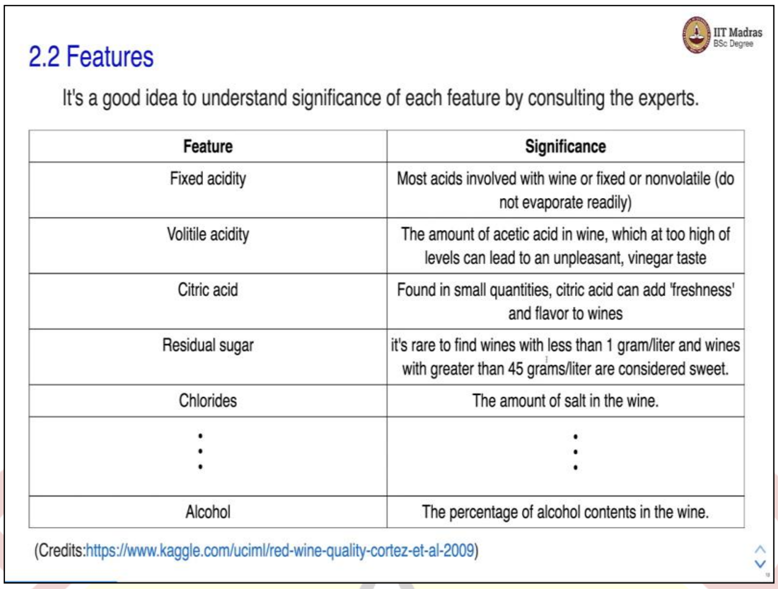
```
data.head()
```

	fixed acidity	volatile acidity	citric acid	residual sugar	chlorides	free sulfur dioxide	total sulfur dioxide	density	pH	sulphates	alcohol	quality
0	7.4	0.70	0.00	1.9	0.076	11.0	34.0	0.9978	3.51	0.56	9.4	5
1	7.8	0.88	0.00	2.6	0.098	25.0	67.0	0.9968	3.20	0.68	9.8	5
2	7.8	0.76	0.04	2.3	0.092	15.0	54.0	0.9970	3.26	0.65	9.8	5
3	11.2	0.28	0.56	1.9	0.075	17.0	60.0	0.9980	3.16	0.58	9.8	6
4	7.4	0.70	0.00	1.9	0.076	11.0	34.0	0.9978	3.51	0.56	9.4	5

We look at the first few data samples with the head method. We simply call the head method on the data frame. The head method gives us the first few rows or exactly five rows from the data frame. Here on your screen, you see the first five rows in this data frame the first row here shows the names of the columns of the data frame. In this case, there are 12 columns fixed acidity volatile acidity, citric acid, residual sugar, chlorides free sulfur dioxide, total sulfur dioxide density, pH, sulfates, alcohol, and quality.

And the first row shows the values for different features for the first example. The second row has the values for each of these features for the second example and the quality is the label that we are interested in predicting. So, this is the feature vector consisting of 11 features and this is the label y for the first example or specifically, y1 is a label for the second example third example fourth example, and fifth example and there are corresponding feature vectors that you can see on your screen.

(Refer Slide Time: 10:03)



2.2 Features

It's a good idea to understand significance of each feature by consulting the experts.

Feature	Significance
Fixed acidity	Most acids involved with wine or fixed or nonvolatile (do not evaporate readily)
Volitile acidity	The amount of acetic acid in wine, which at too high of levels can lead to an unpleasant, vinegar taste
Citric acid	Found in small quantities, citric acid can add 'freshness' and flavor to wines
Residual sugar	it's rare to find wines with less than 1 gram/liter and wines with greater than 45 grams/liter are considered sweet.
Chlorides	The amount of salt in the wine.
⋮	⋮
Alcohol	The percentage of alcohol contents in the wine.

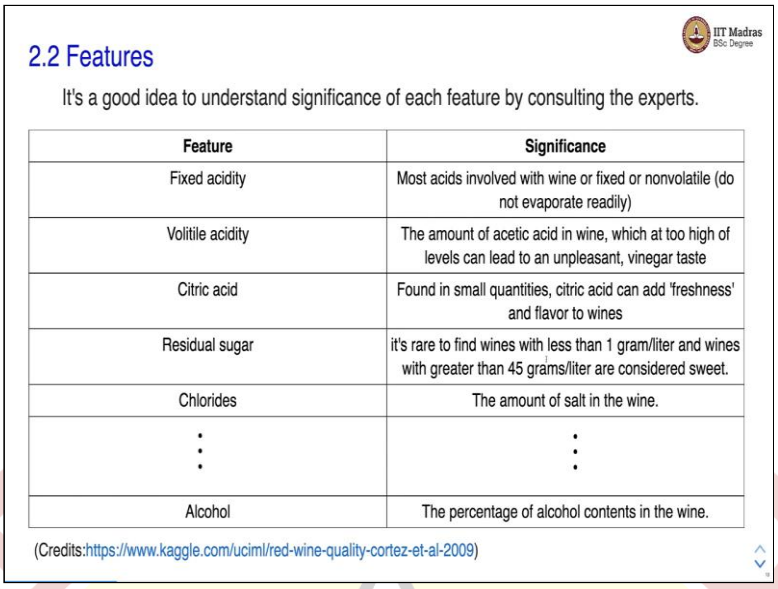
(Credits:https://www.kaggle.com/uciml/red-wine-quality-cortez-et-al-2009)

Once you have data it is a good idea to understand the significance of each feature by consulting the experts. In this case, we have 11 features. The first feature is fixed acidity the significance of this feature is that it captures most acids involved with the wine are fixed or non-volatile. These acids do not evaporate readily then there is something the second feature is volatile acidity.

It captures the amount of acetic acid in wine which are too high of a level can lead to unpleasant or vinegar test in the wine. The third feature is citric acid that captures. So, citric acid is usually found in small quantities in wine and citric acid can add freshness and flavor to the wine. The next feature is residual sugar it is rare to find a wine with less than one gram per liter residual sugar and wines with greater than 45 grams per liter are considered sweet.

Then chlorides which is the amount of salt in the vine and finally alcohol which is the percentage of alcohol content in the wine. So, whenever you are solving a machine learning problem and once you get data it is a good idea to document the features and their significance by consulting the experts.

(Refer Slide Time: 11:43)



2.2 Features

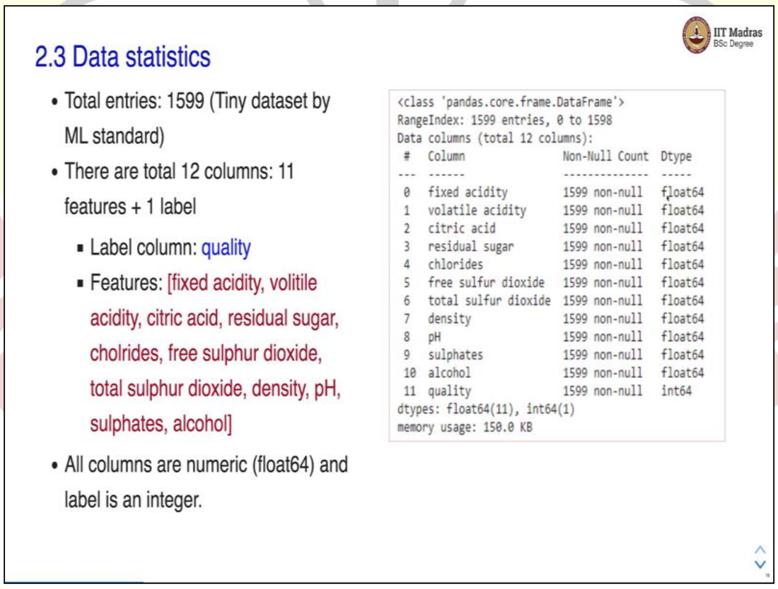
It's a good idea to understand significance of each feature by consulting the experts.

Feature	Significance
Fixed acidity	Most acids involved with wine or fixed or nonvolatile (do not evaporate readily)
Volitile acidity	The amount of acetic acid in wine, which at too high of levels can lead to an unpleasant, vinegar taste
Citric acid	Found in small quantities, citric acid can add 'freshness' and flavor to wines
Residual sugar	it's rare to find wines with less than 1 gram/liter and wines with greater than 45 grams/liter are considered sweet.
Chlorides	The amount of salt in the wine.
⋮	⋮
Alcohol	The percentage of alcohol contents in the wine.

(Credits:https://www.kaggle.com/uciml/red-wine-quality-cortez-et-al-2009)

So, you can access the feature list from this data accessing columns of the data frame. We print values of all columns but the last, the last column is the label. So, you can see 11 features in the feature list and one label, and the feature list has got features like fixed acidity, volatile acidity, citric acid, residual sugar chlorides, free sulfur dioxide, total sulfur dioxide, density, pH, sulfates, and alcohol content and quality is the label that we wish to predict.

(Refer Slide Time: 12:38)



2.3 Data statistics

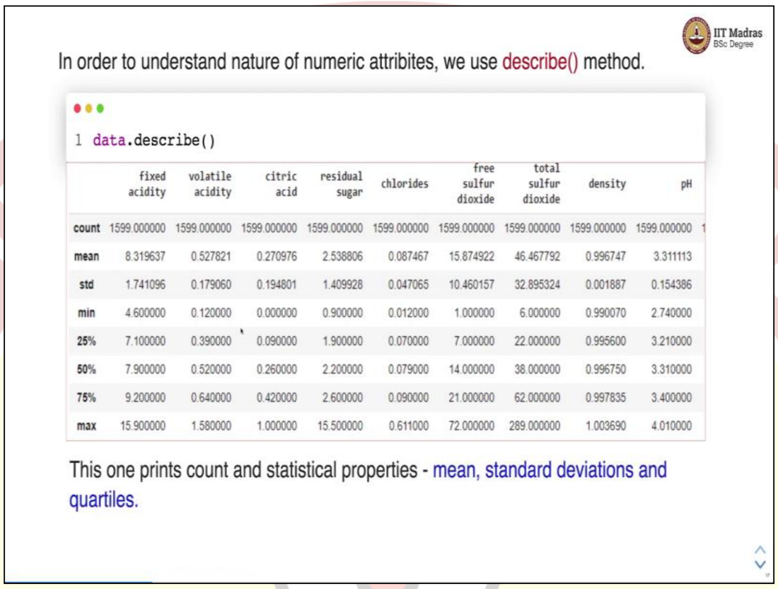
- Total entries: 1599 (Tiny dataset by ML standard)
- There are total 12 columns: 11 features + 1 label
  - Label column: quality
  - Features: [fixed acidity, volitile acidity, citric acid, residual sugar, cholrides, free sulphur dioxide, total sulphur dioxide, density, pH, sulphates, alcohol]
- All columns are numeric (float64) and label is an integer.

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1599 entries, 0 to 1598
Data columns (total 12 columns):
 #   Column                Non-Null Count  Dtype
---  ------                --------------  -----
 0   fixed acidity         1599 non-null   float64
 1   volatile acidity      1599 non-null   float64
 2   citric acid           1599 non-null   float64
 3   residual sugar        1599 non-null   float64
 4   chlorides             1599 non-null   float64
 5   free sulfur dioxide   1599 non-null   float64
 6   total sulfur dioxide  1599 non-null   float64
 7   density               1599 non-null   float64
 8   pH                    1599 non-null   float64
 9   sulphates             1599 non-null   float64
 10  alcohol               1599 non-null   float64
 11  quality               1599 non-null   int64
dtypes: float64(11), int64(1)
memory usage: 150.0 KB
```

Now that we have we have got a fair bit of understanding of the features. Let us look at the data statistics. We use info method on the data frame to get the quick description of the data. So, this particular description talks about various statistics about the data there are 1599 total entries. So, in the by going by ml standard this is really a tiny dataset. There are 12 columns there are 11 features and one label. The label column is quality and the remaining 11 columns are the features.

And you can see that all columns are numeric while the quality or the quality which is a label is an integer column.

(Refer Slide Time: 13:50)



IIT Madras
BSc Degree

In order to understand nature of numeric attribites, we use describe() method.

```
data.describe()
```

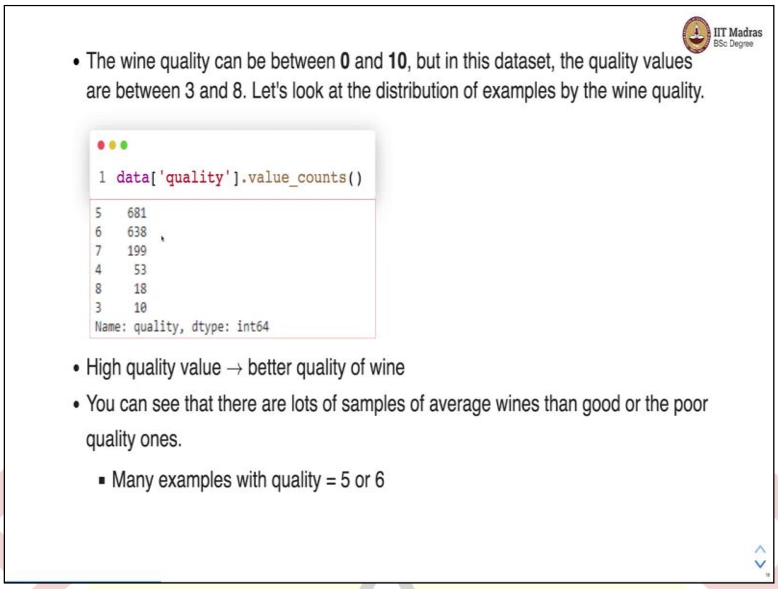
	fixed acidity	volatile acidity	citric acid	residual sugar	chlorides	free sulfur dioxide	total sulfur dioxide	density	pH
count	1599.000000	1599.000000	1599.000000	1599.000000	1599.000000	1599.000000	1599.000000	1599.000000	1599.000000
mean	8.319637	0.527821	0.270976	2.538806	0.087467	15.874922	46.467792	0.996747	3.311113
std	1.741096	0.179060	0.194801	1.409928	0.047065	10.460157	32.895324	0.001887	0.154386
min	4.600000	0.120000	0.000000	0.900000	0.012000	1.000000	6.000000	0.990070	2.740000
25%	7.100000	0.390000	0.090000	1.900000	0.070000	7.000000	22.000000	0.995600	3.210000
50%	7.900000	0.520000	0.260000	2.200000	0.079000	14.000000	38.000000	0.996750	3.310000
75%	9.200000	0.640000	0.420000	2.600000	0.090000	21.000000	62.000000	0.997835	3.400000
max	15.900000	1.580000	1.000000	15.500000	0.611000	72.000000	289.000000	1.003690	4.010000

This one prints count and statistical properties - mean, standard deviations and quartiles.

In order to understand the nature of numeric attributes, we use the described method. The described method gives us statistics about each feature or specifically it gives statistics about each column in the data frame. So, here you can see the statistics about fixed acidity this is a number of examples for this particular feature and the mean is 8.3 with a standard deviation of 1.74.

The minimum value of fixed acidity is 4.6 and the maximum value is 15.9 the 25 percentile is 7.1 58 percentiles is 7.9 and the 75 percentile is 9.2. And you can similarly you know look at the statistics for other features. So, as we just discussed it prints counts and other statistical properties like mean, standard deviation, and quartiles.

(Refer Slide Time: 15:06)


- The wine quality can be between **0** and **10**, but in this dataset, the quality values are between 3 and 8. Let's look at the distribution of examples by the wine quality.

```
data['quality'].value_counts()
```

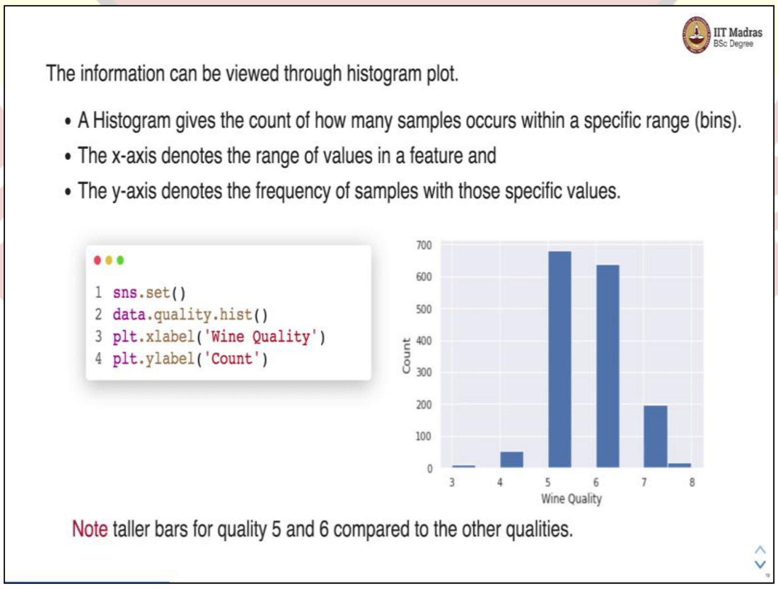
```
5    681
6    638
7    199
4     53
8     18
3     10
Name: quality, dtype: int64
```

- High quality value → better quality of wine
- You can see that there are lots of samples of average wines than good or the poor quality ones.
  - Many examples with quality = 5 or 6

The wine quality can be between 0 and 10 but in this dataset, the wine qualities are between 3 and 8. Let us look at the description. Let us look at the distribution of examples by wine quality. So, we simply call the value counts on the quality column of the data frame and you can see that the wines of quality 5 and 6 are dominating the dataset. The higher quality value implies the better quality of the wine.

And we can see that there are lots of samples of average wines than good or poor quality ones. There are many examples with qualities five and six.

(Refer Slide Time: 16:06)


The information can be viewed through histogram plot.

- A Histogram gives the count of how many samples occurs within a specific range (bins).
- The x-axis denotes the range of values in a feature and
- The y-axis denotes the frequency of samples with those specific values.

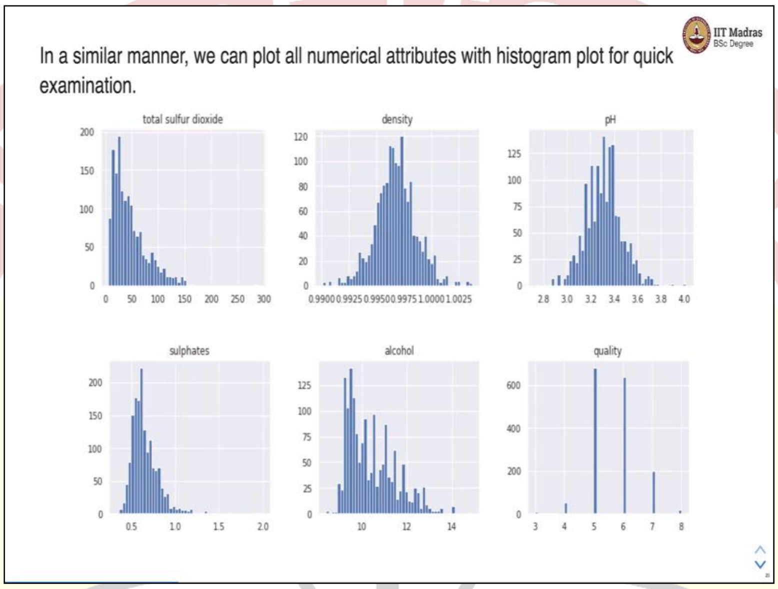
```
sns.set()
data.quality.hist()
plt.xlabel('Wine Quality')
plt.ylabel('Count')
```

Note taller bars for quality 5 and 6 compared to the other qualities.

This information can be viewed through a histogram plot. A histogram gives the count of how many samples occur within a specific range. We use sns as sns or c bond as a plotting library. So, you can see that twine quality is on the x-axis and the count is on the y axis and we have plotted a histogram of the wine quality. In this histogram, you can see that the wines of quality 5 and 6 are far more than the other wines' other vines of different qualities.

For example, wines of quality three are much lesser. Similarly, the wine of quality 8 is also far fewer compared to these two wine types.

(Refer Slide Time: 17:17)

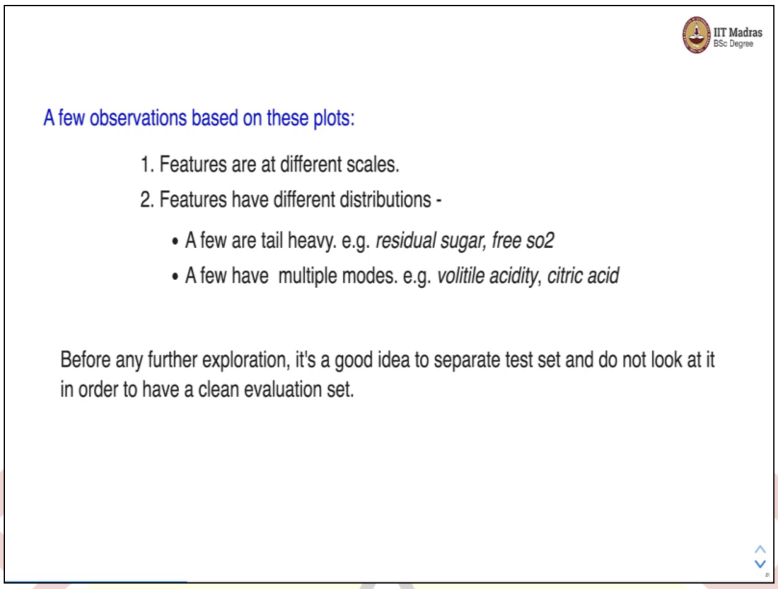


In a similar manner, we can plot all numerical attributes with histogram plot for quick examination.

In a similar manner, we can plot all numerical attributes with histogram plots. So, you can see histogram plots of our different features here and we have also included the histogram plot of the quality for your reference. The x-axis in this plot shows the value of the corresponding features and the y-axis is the count in that particular bin. So, the values of the features are divided into a fixed number of bins and we count the number of values in each bin which is on the y-axis.

So, you can see that different properties or different features have different scales their values vary. So, for example, here the value of total sulphur dioxide can be between 0 to 300. In this case, the range of the total sulfur dioxide is between 0 to 300 whereas the density is between 0.99 and 1. Whereas pH varies from 2.8 to 4 sulfate varies from 0.5 to 2 and alcohol varies from maybe 0 to 14

(Refer Slide Time: 18:49)



IIT Madras
BSc Degree

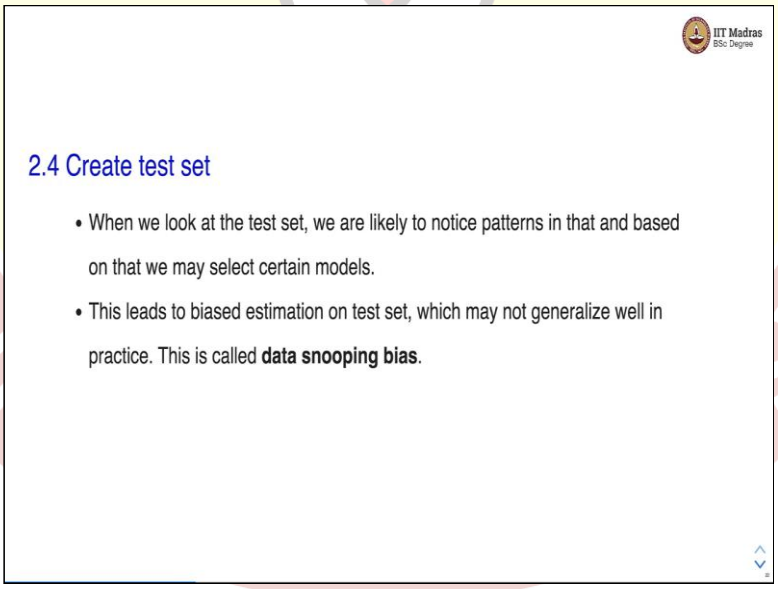
A few observations based on these plots:

1. Features are at different scales.
2. Features have different distributions -
    - A few are tail heavy. e.g. *residual sugar, free so2*
    - A few have multiple modes. e.g. *volitile acidity, citric acid*

Before any further exploration, it's a good idea to separate test set and do not look at it in order to have a clean evaluation set.

So, features different features are at different scales and features have different distributions. Some of them are tail heavy like residual sugar and free SO 2 and a few have multiple modes just like volatile acidity and citric acid. Before any further explanation, it is a good idea to separate the test set and do not look at it in order to have a clean evaluation set.

(Refer Slide Time: 19:19)



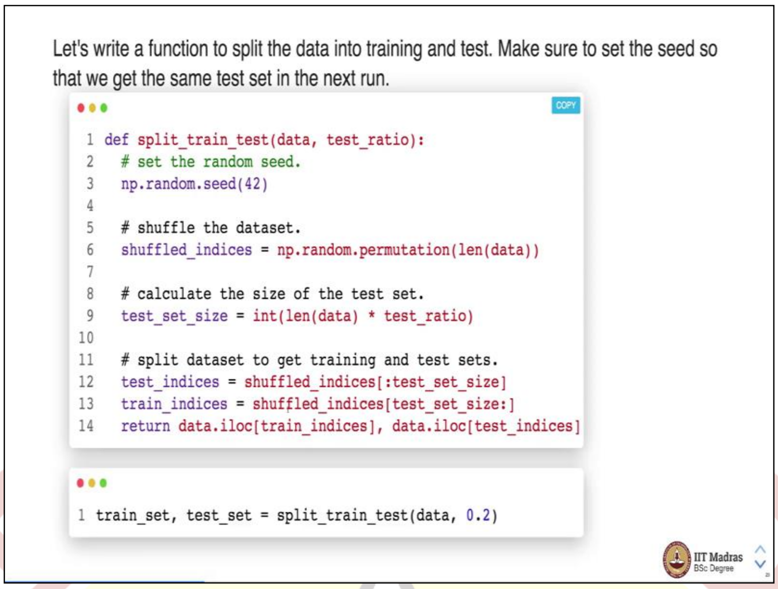
IIT Madras
BSc Degree

2.4 Create test set

- When we look at the test set, we are likely to notice patterns in that and based on that we may select certain models.
- This leads to biased estimation on test set, which may not generalize well in practice. This is called **data snooping bias**.

When we look at the test set we are likely to notice patterns in that and based on that we may select certain models. This leads to biased estimation on the test set which may not generalize well in practice this is called data snooping bias.

(Refer Slide Time: 19:39)



Let's write a function to split the data into training and test. Make sure to set the seed so that we get the same test set in the next run.

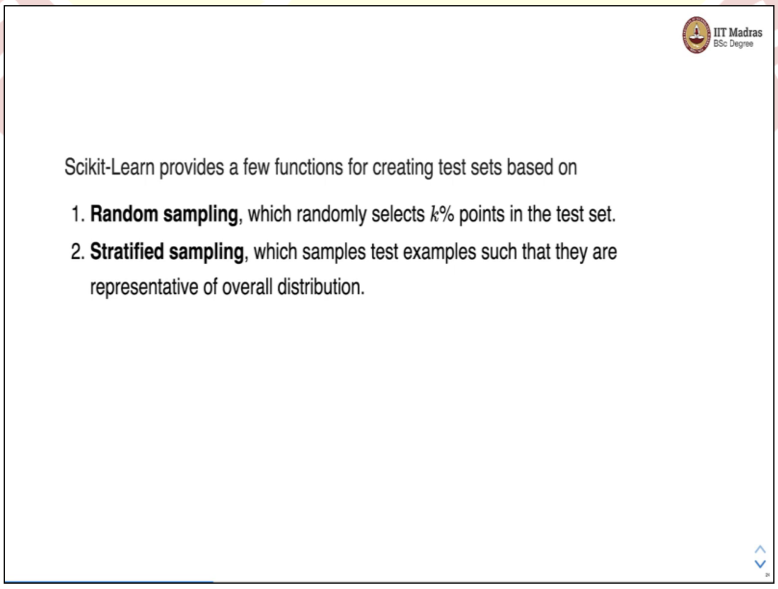
```
def split_train_test(data, test_ratio):
  # set the random seed.
  np.random.seed(42)

  # shuffle the dataset.
  shuffled_indices = np.random.permutation(len(data))

  # calculate the size of the test set.
  test_set_size = int(len(data) * test_ratio)

  # split dataset to get training and test sets.
  test_indices = shuffled_indices[:test_set_size]
  train_indices = shuffled_indices[test_set_size:]
  return data.iloc[train_indices], data.iloc[test_indices]
```

```
train_set, test_set = split_train_test(data, 0.2)
```

Let us write a function to test the data, let us write the function to speed the data into training and test. Make sure to set the seed so that we get the same test set in in the next run and in subsequent runs you know after this. So, in this particular function, we take the data and the ratio of test examples. We first set the random seed and here we have set it to 42 then we shuffle the data set then we calculate the size of the test set and we split the data set to get training and test sets.

So, the data set is shuffled with np dot random dot permutation method and then the shuffled indices are used to split the data into training and test. So, in this case, we have called the split trend test method with the test ratio of 0.2 which means 80 examples will be used as training and 20 examples will be set aside for testing.

(Refer Slide Time: 21:01)

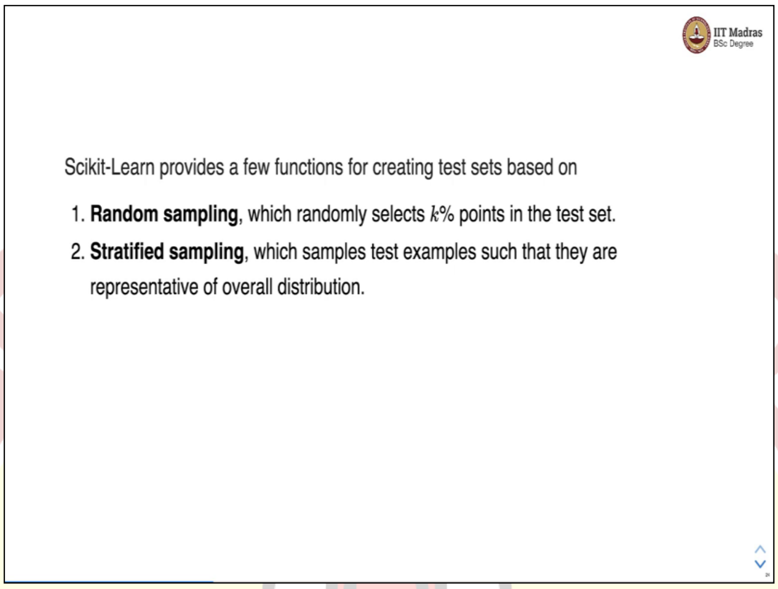


Scikit-Learn provides a few functions for creating test sets based on

1. **Random sampling**, which randomly selects k% points in the test set.
2. **Stratified sampling**, which samples test examples such that they are representative of overall distribution.

Scikit-learn provides a few functions for creating test sets based on random sampling which randomly selects k percentage point in the test set or the stratified sampling with samples to test examples such that they are representative of overall distribution.

(Refer Slide Time: 21:22)



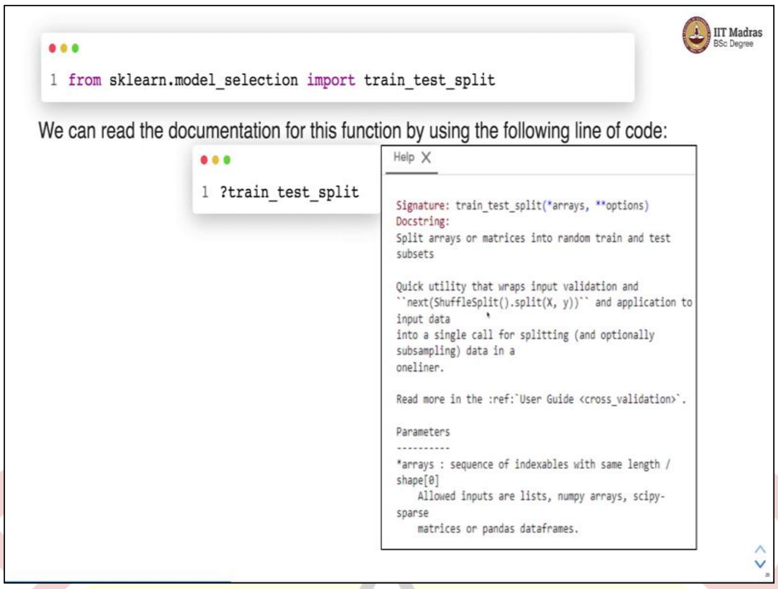
Scikit-Learn provides a few functions for creating test sets based on

1. **Random sampling**, which randomly selects k% points in the test set.
2. **Stratified sampling**, which samples test examples such that they are representative of overall distribution.

The trend test split function performs random sampling with random state parameters to set the random seed which ensures that the same examples are selected for test runs across different runs. The test size parameter for specifying the size of the test set. The shuffle flag to specify if the data needs to be shuffled before splitting. It also has provisions for processing multiple data sets with an identical number of rows and selecting the indices from those datasets.

This is useful when the features and labels are in different data frames. So, the trend test split function is in the sklearn dot model selection library. So, we import it from a sklearn dot model selection import train underscore test under underscore split.

(Refer Slide Time: 22:33)


IIT Madras
BSc Degree

```python
from sklearn.model_selection import train_test_split
```

We can read the documentation for this function by using the following line of code:

```python
?train_test_split
```

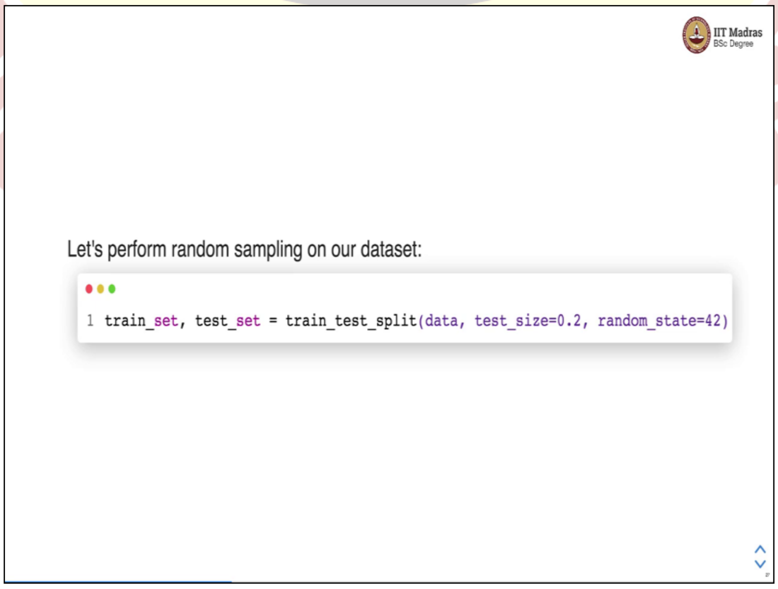
Help X

```
Signature: train_test_split(*arrays, **options)
Docstring:
Split arrays or matrices into random train and test
subsets

Quick utility that wraps input validation and
``next(ShuffleSplit().split(X, y))`` and application to
input data
into a single call for splitting (and optionally
subsampling) data in a
oneliner.

Read more in the :ref:`User Guide <cross_validation>`.

Parameters
----------
*arrays : sequence of indexables with same length /
shape[0]
    Allowed inputs are lists, numpy arrays, scipy-
sparse
    matrices or pandas dataframes.
```

So, whenever we come across a new function there is a good chance that we are not aware of all the parameters. While we have listed some of the key parameters of this particular function there may be some more parameters that you might require in certain typical situations. So, you can use a question mark followed by the name of the function in order to read the documentation of this function.

So, before calling for the documentation make sure that you have imported the imported API or the function from the sklearn library. So, when you access when you try to access the document with the question mark command the document opens and it shows a different kinds of parameters and the docs string of this particular function.

(Refer Slide Time: 23:40)

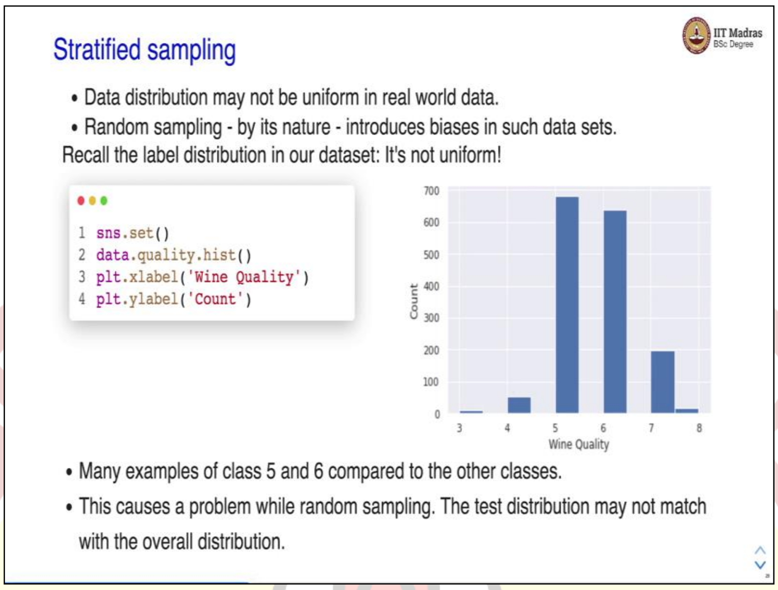

IIT Madras
BSc Degree

Let's perform random sampling on our dataset:

```python
train_set, test_set = train_test_split(data, test_size=0.2, random_state=42)
```

So, let us apply random sampling to our data set to split it into two sets training and test. Here it is we set aside 20 examples for the test set.

(Refer Slide Time: 24:00)



Stratified sampling

- Data distribution may not be uniform in real world data.
- Random sampling - by its nature - introduces biases in such data sets.

Recall the label distribution in our dataset: It's not uniform!

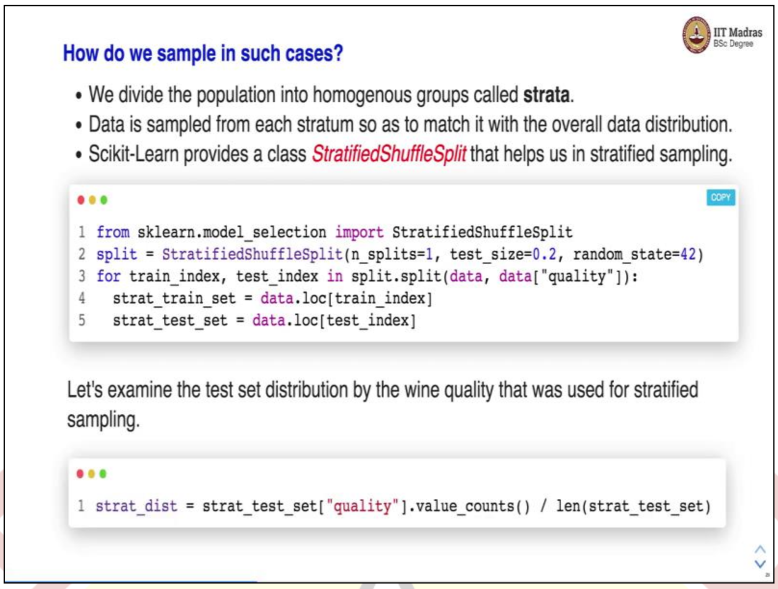
```
sns.set()
data.quality.hist()
plt.xlabel('Wine Quality')
plt.ylabel('Count')
```

- Many examples of class 5 and 6 compared to the other classes.
- This causes a problem while random sampling. The test distribution may not match with the overall distribution.

So, whenever the data distribution is not uniform we use what is called stratified sampling. Random sampling introduces biases in the non-uniform datasets. So, in our wine quality dataset, the label distribution in our dataset is not uniform. The labels of types 5 and 6 are far more represented than the other types of labels. In this case, labels 5 and 6 are overrepresented in the data sets whereas other labels are underrepresented in the dataset.

So, the distribution of the label is not uniform and this could be a problem for random sampling and the problem would imply that the test distribution may not match the overall distribution.

(Refer Slide Time: 25:12)



How do we sample in such cases?

- We divide the population into homogenous groups called **strata**.
- Data is sampled from each stratum so as to match it with the overall data distribution.
- Scikit-Learn provides a class *StratifiedShuffleSplit* that helps us in stratified sampling.

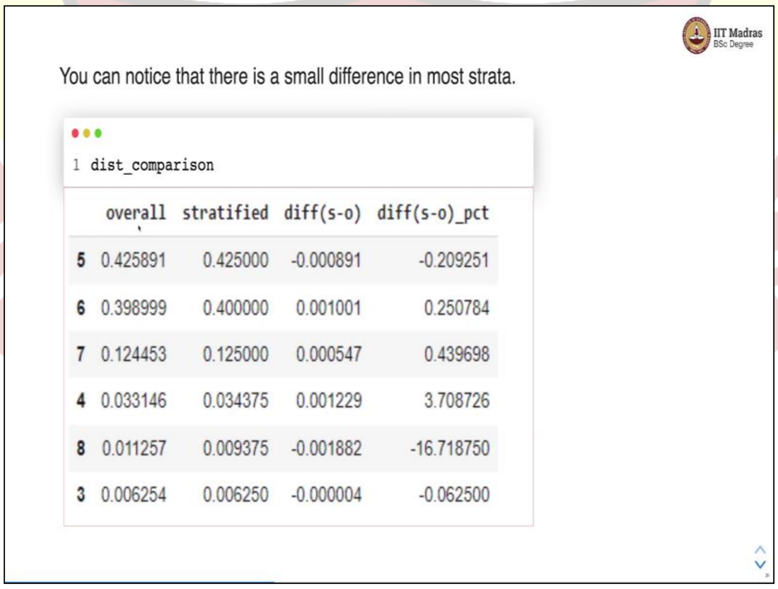
```
from sklearn.model_selection import StratifiedShuffleSplit
split = StratifiedShuffleSplit(n_splits=1, test_size=0.2, random_state=42)
for train_index, test_index in split.split(data, data["quality"]):
  strat_train_set = data.loc[train_index]
  strat_test_set = data.loc[test_index]
```

Let's examine the test set distribution by the wine quality that was used for stratified sampling.

```
strat_dist = strat_test_set["quality"].value_counts() / len(strat_test_set)
```

So, how do we sample in such cases? So, we divide the population into homogeneous groups called strata. Data is sampled from each stratum so as to match it with the overall data distribution. Scikit-learn provides a stratified shuffle split class that helps us in stratified sampling. So, the stratified shuffle split is also present in the sklearn dot model selection. We need to provide the number of splits the size of the test set and the random state just like the train and test trend test split function. Let us examine the tested distribution of the wine quality that was used for stratified sampling.

(Refer Slide Time: 26:07)



You can notice that there is a small difference in most strata.

```
dist_comparison
```

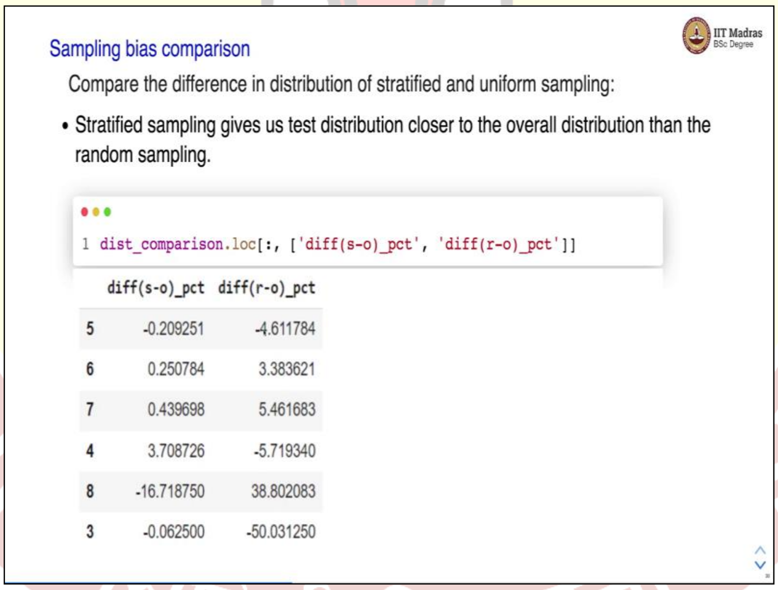
	overall	stratified	diff(s-o)	diff(s-o)_pct
5	0.425891	0.425000	-0.000891	-0.209251
6	0.398999	0.400000	0.001001	0.250784
7	0.124453	0.125000	0.000547	0.439698
4	0.033146	0.034375	0.001229	3.708726
8	0.011257	0.009375	-0.001882	-16.718750
3	0.006254	0.006250	-0.000004	-0.062500

And we will compare it with the overall distribution. Let us look at them side by side and we can notice that there is a small difference in most of the strata. So, in this table, we have the overall distribution of different wine types. So, these are the wine types one type 5, 6, 7, 4, 8, and 3 this is the overall distribution in the training set 42% of examples come from the type 5, 39% examples come from type 6, 12% examples come from type 7, 3% comes from type 4, 1% from type 8 and less than 1% t or 0.6% examples come from type 3.

And when we perform stratified sampling you can see that the overall distribution of the data is maintained. Here as against 0.42 or 42% examples of type 5 in stratified sampling, we get 42% examples from type 5. In other cases, also you can see that the overall and stratified numbers are very close. So, in the third column, we have you know represented the difference between stratified and overall distribution.

And the last column provides what is the percentage difference between the overall and stratified cases. So, the difference is pretty small and you can see that even in terms of percentage the difference is minuscule in many cases except for this particular case where the difference is about 16.71%.

(Refer Slide Time: 28:08)



Sampling bias comparison

Compare the difference in distribution of stratified and uniform sampling:

- Stratified sampling gives us test distribution closer to the overall distribution than the random sampling.

```
dist_comparison.loc[:, ['diff(s-o)_pct', 'diff(r-o)_pct']]
```

	diff(s-o)_pct	diff(r-o)_pct
5	-0.209251	-4.611784
6	0.250784	3.383621
7	0.439698	5.461683
4	3.708726	-5.719340
8	-16.718750	38.802083
3	-0.062500	-50.031250

Let us compare this with random sampling. So, when we compared the overall distribution and distribution obtained from the random sample you can see that there is a large difference. So, this is this difference is about 4.6%, 3% 5.4% this is 38, 39% difference and about 50% difference. So, you can see that in smaller classes the differences are far more compared to larger classes.

So, you can see that the stratified sampling gives us a test distribution that is closely matching to the overall distribution as compared to the random sample.