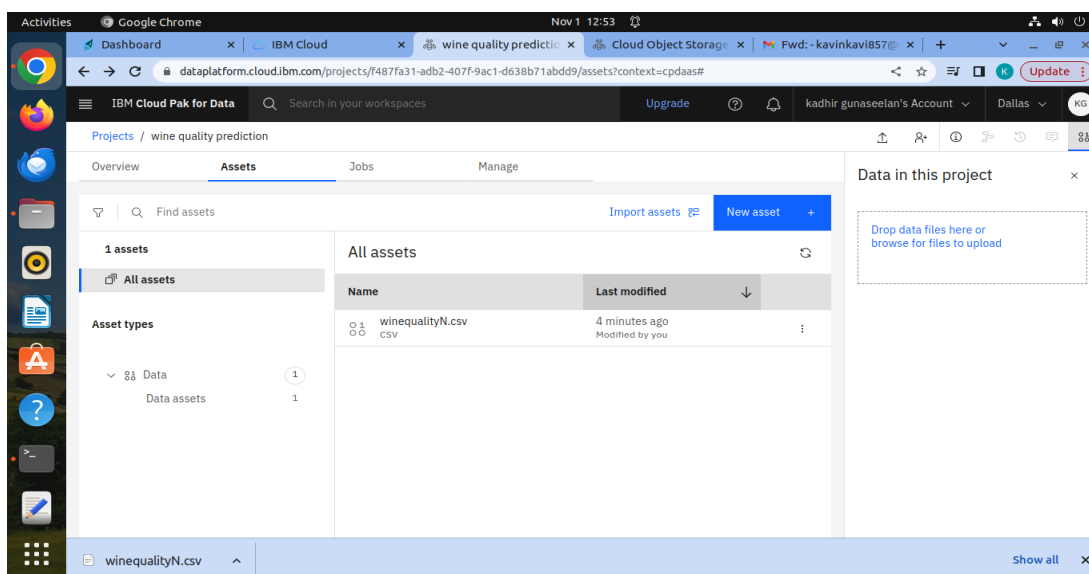


# Development phase - 1

## Machine learning model deployment using IBM Watson Studio

### Wine quality prediction

Here we will predict the quality of wine on the basis of given features. We use the wine quality dataset available. This dataset has the fundamental features which are responsible for affecting the quality of the wine. By the use of several Machine learning models, we will predict the quality of the wine.



The screenshot shows the 'Preview asset' tab for the 'winequalityN.csv' asset. The preview displays a table with 13 columns and 1000 rows. The columns are: type, fixed acidity, volatile acid..., citric acid, residual su..., chlorides, free sulfur diox..., and total sulf. The table shows 10 rows of data.

type	fixed acidity	volatile acid...	citric acid	residual su...	chlorides	free sulfur diox...	total sulf
white	7	0.27	0.36	20.7	0.045	45	170
white	6.3	0.3	0.34	1.6	0.049	14	132
white	8.1	0.28	0.4	6.9	0.05	30	97
white	7.2	0.23	0.32	8.5	0.058	47	186
white	7.2	0.23	0.32	8.5	0.058	47	186
white	8.1	0.28	0.4	6.9	0.05	30	97
white	6.2	0.32	0.16	7	0.045	30	136
white	7	0.27	0.36	20.7	0.045	45	170
white	6.3	0.3	0.34	1.6	0.049	14	132
white	8.1	0.22	0.43	1.5	0.044	28	129

# Importing libraries and Dataset:

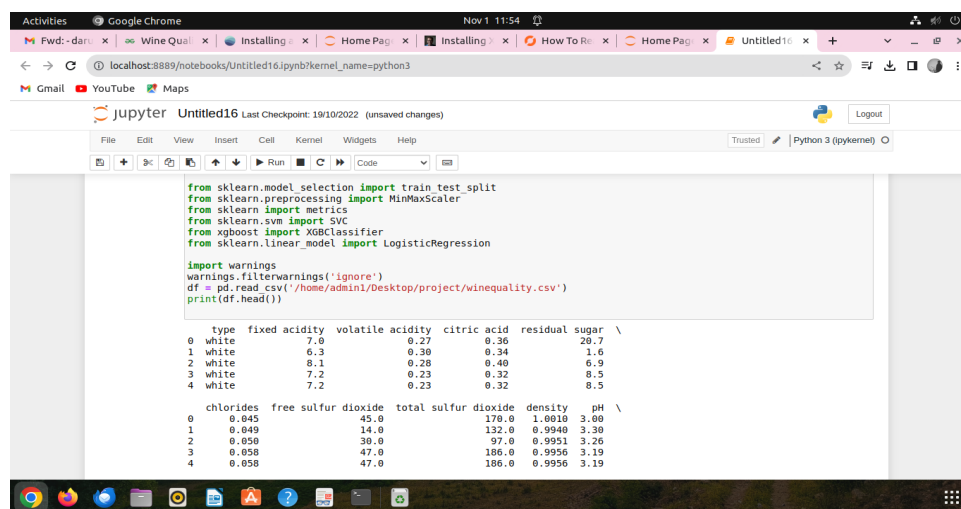
- **Pandas** is a useful library in data handling.
- **Numpy** library used for working with arrays.
- **Seaborn/Matplotlib** are used for data visualisation purposes.
- Sklearn – This module contains multiple libraries having pre-implemented functions to perform tasks from data preprocessing to model development and evaluation.
- **XGBoost** – This contains the eXtreme Gradient Boosting machine learning algorithm which is one of the algorithms which helps us to achieve high accuracy on prediction.

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

```
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import MinMaxScaler
from sklearn import metrics
from sklearn.svm import SVC
from xgboost import XGBClassifier
from sklearn.linear_model import LogisticRegression
```

```
import warnings
warnings.filterwarnings('ignore')
```

Now let's look at the first five rows of the dataset.



```
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import MinMaxScaler
from sklearn import metrics
from sklearn.svm import SVC
from xgboost import XGBClassifier
from sklearn.linear_model import LogisticRegression

import warnings
warnings.filterwarnings('ignore')
df = pd.read_csv('/home/admin1/Desktop/project/winequality.csv')
print(df.head())
```

	type	fixed acidity	volatile acidity	citric acid	residual sugar	
0	white	7.0	0.27	0.36		20.7
1	white	6.3	0.30	0.34		1.6
2	white	8.1	0.28	0.40		6.9
3	white	7.2	0.23	0.32		8.5
4	white	7.2	0.23	0.32		8.5

	chlorides	free sulfur dioxide	total sulfur dioxide	density	pH	
0	0.045	45.0	170.0	1.0010	3.00	
1	0.049	14.0	132.0	0.9940	3.30	
2	0.050	30.0	97.0	0.9951	3.26	
3	0.058	47.0	186.0	0.9956	3.19	
4	0.058	47.0	186.0	0.9956	3.19	

Let's explore the type of data present in each of the columns present in the dataset.

```
df = pd.read_csv('winequality.csv')  
print(df.head())
```

```
In [2]: df.info()  
  
<class 'pandas.core.frame.DataFrame'>  
RangeIndex: 6497 entries, 0 to 6496  
Data columns (total 13 columns):  
#   Column                Non-Null Count  Dtype    
---  --    
0    type                  6497 non-null  object   
1    fixed acidity          6487 non-null  float64  
2    volatile acidity       6489 non-null  float64  
3    citric acid            6494 non-null  float64  
4    residual sugar         6495 non-null  float64  
5    chlorides              6495 non-null  float64  
6    free sulfur dioxide     6497 non-null  float64  
7    total sulfur dioxide    6497 non-null  float64  
8    density                6497 non-null  float64  
9    pH                    6488 non-null  float64  
10   sulphates              6493 non-null  float64  
11   alcohol                6497 non-null  float64  
12   quality                6497 non-null  int64    
dtypes: float64(11), int64(1), object(1)  
memory usage: 660.0+ KB
```

Now we'll explore the descriptive statistical measures of the dataset.

```
df.describe().T
```

```
In [3]: df.describe().T  
  
Out[3]:  


|                      | count  | mean       | std       | min     | 25%      | 50%       | 75%       | max       |
|----------------------|--------|------------|-----------|---------|----------|-----------|-----------|-----------|
| fixed acidity        | 6487.0 | 7.216579   | 1.296750  | 3.80000 | 6.40000  | 7.00000   | 7.70000   | 15.90000  |
| volatile acidity     | 6489.0 | 0.339691   | 0.164649  | 0.08000 | 0.23000  | 0.29000   | 0.40000   | 1.58000   |
| citric acid          | 6494.0 | 0.318722   | 0.145265  | 0.00000 | 0.25000  | 0.31000   | 0.39000   | 1.66000   |
| residual sugar       | 6495.0 | 5.444326   | 4.758125  | 0.60000 | 1.80000  | 3.00000   | 8.10000   | 65.80000  |
| chlorides            | 6495.0 | 0.056042   | 0.035036  | 0.00900 | 0.03800  | 0.04700   | 0.06500   | 0.61100   |
| free sulfur dioxide  | 6497.0 | 30.525319  | 17.749400 | 1.00000 | 17.00000 | 29.00000  | 41.00000  | 289.00000 |
| total sulfur dioxide | 6497.0 | 115.744574 | 56.521855 | 6.00000 | 77.00000 | 118.00000 | 156.00000 | 440.00000 |
| density              | 6497.0 | 0.994697   | 0.002999  | 0.98711 | 0.99234  | 0.99489   | 0.99699   | 1.03898   |
| pH                   | 6488.0 | 3.218395   | 0.160748  | 2.72000 | 3.11000  | 3.21000   | 3.32000   | 4.01000   |
| sulphates            | 6493.0 | 0.531215   | 0.148814  | 0.22000 | 0.43000  | 0.51000   | 0.60000   | 2.00000   |
| alcohol              | 6497.0 | 10.491801  | 1.192712  | 8.00000 | 9.50000  | 10.30000  | 11.30000  | 14.90000  |
| quality              | 6497.0 | 5.818378   | 0.873255  | 3.00000 | 5.00000  | 6.00000   | 6.00000   | 9.00000   |


```