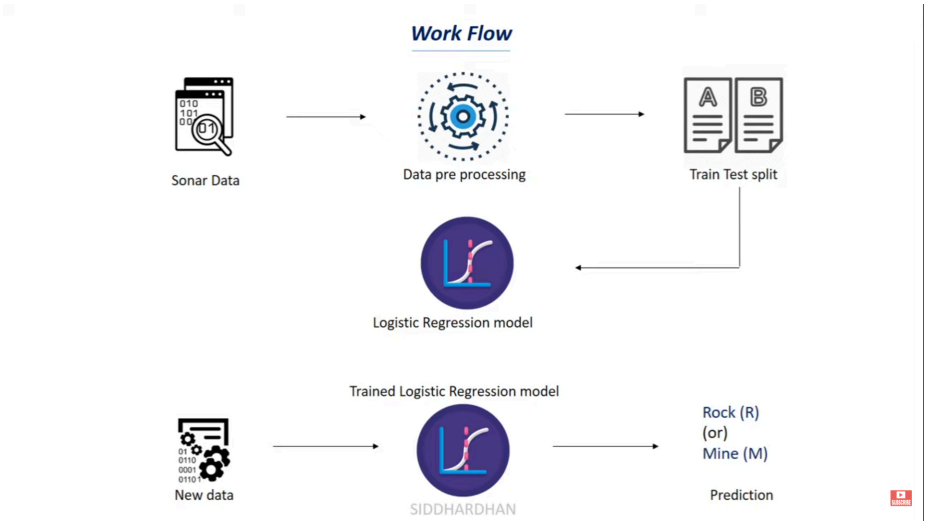
Project 1 - **Sonar Mine vs Rock with Python**



- Machine Learning Model - **Logistic Regression Model**
- Works really well for binary classification (either rock or mine/ 0 or 1)
- Uses Supervised learning Algorithm - A type of machine learning where the model learns from labeled data to make predictions or decisions.

Example Dataset link -
https://drive.google.com/file/d/1pQxtljlNVh0DHYg-Ye7dtpDTlFceHVfa/view
[Extracted from a laboratory setup of a metal cylinder(as mine is a metal) and a rock]

Required Dependencies

```
import numpy as np
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score
```

- numpy as np - for Arrays
- pandas as pd - for certain data processing sets / is used for data manipulation and analysis.
- from sklearn - good library for machine learning algorithms and other functions
    - from sklearn.model_selection

        - train_test_split - function for splitting training and test data so we don't need to do it manually (https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.train_test_split.html)

    - from sklearn.linear_model
        - LogisticRegression - is a machine learning algorithm in scikit-learn used for binary or multiclass classification, where it learns to predict categories based on labeled training data. (https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LogisticRegression.html)
    - from sklearn.metrics
        - accuracy_score - is a function in scikit-learn that calculates the ratio of correct predictions to total predictions made by a model. (https://scikit-learn.org/stable/modules/generated/sklearn.metrics.accuracy_score.html)

Data Collection and Data Processing

- Uploading of data - used .csv file for now, can use APIs to directly call and upload the data.
- Creating a panda dataframe by reading the csv file "sonar_data = pandas.read_csv()". (https://pandas.pydata.org/docs/reference/api/pandas.read_csv.html)
- As there is no header for the csv file, "header=None".
- "sonar_data.shape" - gives the number of rows and column
- "sonar_data.describe()" - the description of the dataframe(count, mean, max, min, 25%, 50%, etc…). #statistical measures
- "sonar_data[60].value_counts()" - to see how many rocks and mines are there.. Taking index as 60 because there 61 columns but python indexing starts from 0.

* For accurate prediction, both should have similar amount of datasets.. Eg: M = 111, R = 97
* If eg: M = 111, R = 27, the prediction won't be accurate.

- "sonar_data.groupby(60).mean()" - gives the mean datasets of R and M. will be used for the prediction.

* For supervised learning labels are used.(R and M) whereas for unsupervised learning, labels are not required.

- X - Dropping the 60th column, which is the labels column and specifying the axis as 1(axis 0 = row, axis 1 = column).
- Y - Assigning the labels column.

Training and test data

- `test_size=0.1` - 10% of the dataset is allocated for testing, and 90% for training.
- `stratify=Y` - Ensures the split maintains the same class distribution in both the training and testing sets.
- `random_state=1` - Sets a seed so the split is reproducible.
- Using the model imported from sklearn.linear_model and training it with the training data set.
- A larger model with larger data set will a huge time.. As this is a small model with a small data set, it will work soon.
- Using sklearn.metrics.accuracy_score, checking the models accuracy after training it with the train data and testing with test data.