by Abhishek Thakur

Arranging machine learning projects

In [1]:
```
# The inside of the project folder should look something like the following.
'''
├── input
│   ├── train.csv
│   └── test.csv
├── src
│   ├── create_folds.py
│   ├── train.py
│   ├── inference.py
│   ├── models.py
│   ├── config.py
│   └── model_dispatcher.py
├── models
│   ├── model_rf.bin
│   └── model_et.bin
├── notebooks
│   ├── exploration.ipynb
│   └── check_data.ipynb
├── README.md
└── LICENSE
'''
```

Out[1]:
```
'\n├── input\n│   ├── train.csv\n│   └── test.csv\n├── src\n│   ├── create_folds.py\n
│   ├── train.py\n│   ├── inference.py\n│   ├── models.py\n│   ├── config.py\n│   └── mod
el_dispatcher.py\n├── models\n│   ├── model_rf.bin\n│   └── model_et.bin\n├── noteb
ooks\n│   ├── exploration.ipynb\n│   └── check_data.ipynb\n├── README.md\n└── LICEN
SE\n'
```

input/: This folder consists of all the input files
src/: We will keep all the python scripts associated with the project here. If I talk about a python script, i.e. any *.py file, it is stored in the src folder.
models/: This folder keeps all the trained models.
notebooks/: All jupyter notebooks (i.e. any *.ipynb file) are stored in the notebooks folder.
README.md: This is a markdown file where you can describe your project and write instructions on how to train the model or to serve this in a production environment.
LICENSE: This is a simple text file that consists of a license for the project, such as MIT, Apache, etc.

If the distribution of labels is quite good and even. We can thus use accuracy/F1 as metrics. This is the first step when approaching a machine learning problem: decide the metric!

In [2]:
```
# src/train.py

import joblib
import pandas as pd
from sklearn import metrics
from sklearn import tree

def run(fold):
    # read the training data with folds
    df = pd.read_csv("/home/hduser/jupyter/winequality-red_n_folds.csv")
    # training data is where kfold is not equal to provided fold
    # also, note that we reset the index
    df_train = df[df['kfold'] != fold].reset_index(drop=True)

    # validation data is where kfold is equal to provided fold
    df_valid = df[df['kfold'] == fold].reset_index(drop=True)

    # drop the label column from dataframe and convert it to
    # a numpy array by using .values.
    # target is label column in the dataframe
    x_train = df_train.drop('quality', axis=1).values
    y_train = df_train['quality'].values

    # similarly, for validation, we have
    x_valid = df_valid.drop("quality", axis=1).values
    y_valid = df_valid['quality'].values

    # initialize simple decision tree classifier from sklearn
    clf = tree.DecisionTreeClassifier()

    # fit the model on training data
    clf.fit(x_train, y_train)

    # create predictions for validation samples
    preds = clf.predict(x_valid)

    # calculate & print accuracy
    accuracy = metrics.accuracy_score(y_valid, preds)
    print(f"Fold={fold}, Accuracy={accuracy}")

    # save the model
    joblib.dump(clf, f"/home/hduser/jupyter/dt_{fold}.bin")


if __name__ == "__main__":
    run(fold=0)
    run(fold=1)
    run(fold=2)
    run(fold=3)
    run(fold=4)
```

```
Fold=0, Accuracy=0.60625
Fold=1, Accuracy=0.603125
Fold=2, Accuracy=0.571875
Fold=3, Accuracy=0.6125
Fold=4, Accuracy=0.5830721003134797
```

You can run this script by calling **python train.py** in the console.

below 5 files are creating in the location
dt_0.bin
dt_1.bin
dt_2.bin
dt_3.bin
dt_4.bin

In [ ]: