Steps and Tasks

1. Import the datasets and libraries, check shape and datatype.

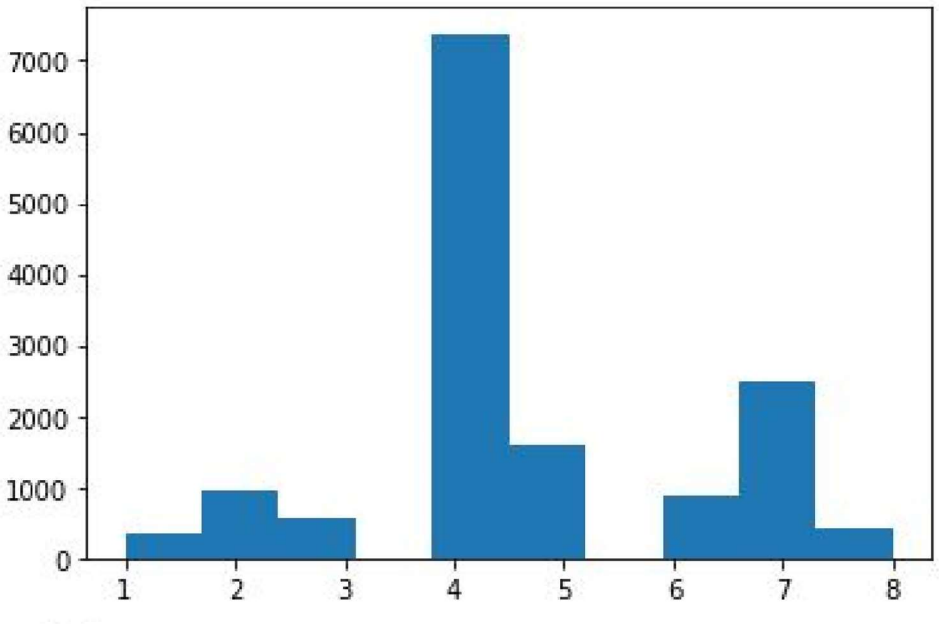
You basically import pre installed python libraries or packages like numpy, pandas, matplotlib and seaborn for data cleaning and visualisation. Scikitlearn and Keras are imported for machine learning models and neural networks. Using the pandas library we can simply import the dataset in csv format as a pandas dataframe. We can then explore the dataset and check the number of features and samples in the data.

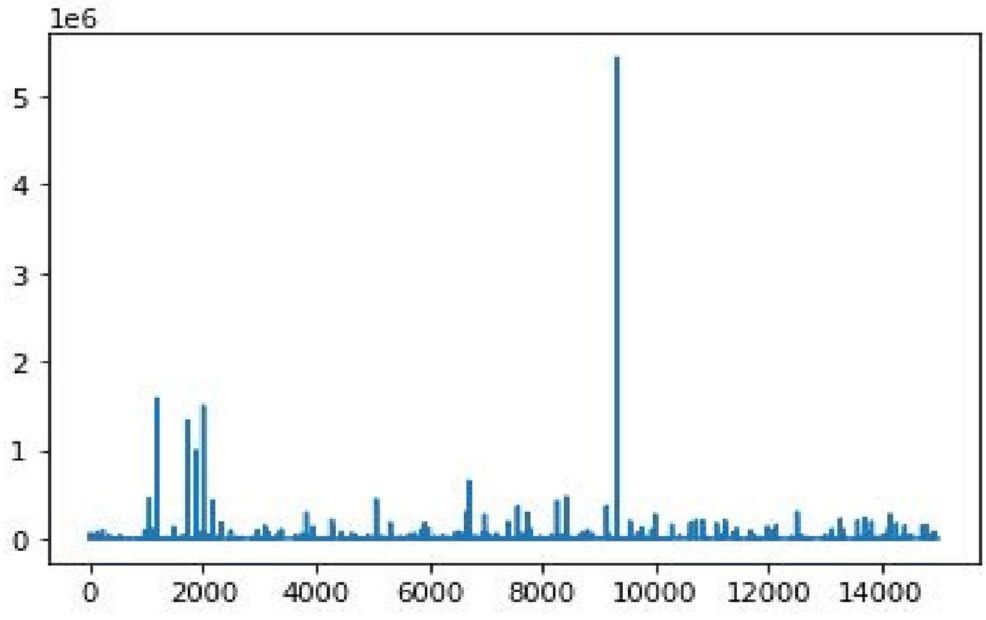
- · Import numpy, pandas, matplotlib and seaborn ● · Import dataset using pandas.csv() method ● · Use data.shape etc.

2. Visualise the dataset using plotting using heatmaps and plots. You can study data distributions for each attribute as well.

Matplotlib and seaborn libraries can be used for plotting. We can plot for each of the features and visually see the distribution of the data with respect to that feature. We can also use it to spot any outliers whatsoever which will help our model to train and learn better. We can plot a heatmap using the seaborn library where we can visualise the correlations of different features with respect to each other. It helps to see how independent are the features because we may want to remove the features which may not add any value to our model.

- · Plot distributions
- · Plot heatmap of correlation values
- · Remove outliers


7000
6000
5000
4000
3000
2000
1000
0
1
2
3
4
5
6
7
8


1e6
5
4
3
2
1
0
0
2000
4000
6000
8000
10000
12000
14000

3. Clean the dataset by removing missing values and other things.

Cleaning the dataset is one of the most essential tasks while solving a machine learning problem. Here in this problem we can remove 'F' and other missing values that might hurt the performance of our model.

- · Remove missing values
- · Remove any other unimportant feature or piece of data

4. Transform attributes into numerical values and other necessary transformations

Machine Learning models need data to be converted to numerical form at the end. We have categorical data and data in other formats which may want to convert. We will use label encoder

and other date & time functions for that task. This part is quite open ended and also known as feature engineering. You can transform original features into new ones which might give better results.

- · Convert categorical data using label encoder
- · Convert into numerical data
- · Feature selection

5. Normalise your data and split the data into training, validation and test set in the appropriate ratio.

We need to split the data into training and test data. We use training data to learn patterns in the data and then check if it generalises well on unseen data. The split percentage can be varied and is generally on the amount of data we have. For a relatively small dataset like this, the split percentage should be high 8020 rather than 991 with respect to train:test). Normalisation is done to ensure all the features all weighted appropriately in the training stage. Just because some features have high scale should not translate to having higher influence on the model. Normalisation can be done using Standard Scalar or MinMax Scalar among others. In this particular problem, MinMax Scalar has been used which basically transforms each variable in the range of 0 to 1.

- · Split dataset in train and test as well as into inputs and outputs ● · Normalise the dataset using scalars

6. Use linear regression, support vector regressor, random forest and for training and get errors.

Different machine learning models can be used for training out of which we can choose whichever gives the best result. We will use the scikit learn libraries for importing these models and the training them use the fit method and providing necessary labelled data (input and output). We are optimising for mean square error here because it's a regression problem after all. The metrics that we can use for us to compare different models can be mean square error and mean absolute error.

- · Import scikit learn library
- · Import the model and define
- · Use .fit method with data as arguments to train
- · Calculate errors

7. Use Decision Tree Regressor and Random Forest Regressors.

Another class of machine learning algorithms include decision trees and random forests. We can import these models from sklearn.tree and then again use the .fit function. We need to give appropriate hyper parameters for them. These are something we can experiment with to achieve better results.

- · Import models
- · Assign hyperparameters for random forest
- · Train the models
- · Calculate errors
-

8. Build an artificial neural network and train it with different layers and hyperparameters. Experiment a little. Use keras.

At the end we can play around with neural networks using the keras library which is quite similar to the scikit learn library. Here we can define the model architecture (layers, number of neurons and activation functions), optimisation algorithm (like gradient descent or adam), cost function (mean square error) and give in the dataset. Then the model trains for different epochs (going through a dataset once means one epoch) to result in an improved model. We may need to perform hyperparameter tuning (i.e. selecting the best hyperparameters like number of neurons or activation function to yield minimum error).

We need to accept the fact that in some cases neural network may not perform better than other machine learning models.

- · Import keras
- · Define model architecture
- · Define optimisation algorithm and cost function
- · Train the neural network
- · Calculate errors

9. Pick the best model based on error as well as generalisation.

Now that we have different machine learning models and their respective errors, we can select the one with minimum error or any other selection scheme. It's important to make sure there is no overfitting and the model generalises well. We use the test data we got from splitting the original dataset (which acts as validation set in our case).

- · Compare errors for each model
- · Select the most suitable ones
- · Use F1 Score and RC Curves Bonus)

10. Save your model and predict on the test set.

Saving the models can be done through both keras and scikit learn. This helps us to load the model again or share it with others. At the end, we use completely unknown data from 'test.csv' to make our final predictions for evaluation from our chosen model in the previous step

- · Save models
- · Make predictions