**Work Force Retention System**

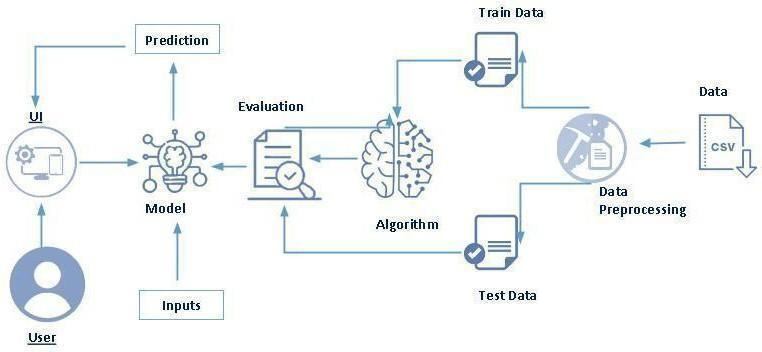
# **Project description:**

The project “Work Force Retention System” involves developing an algorithm that can predict the likelihood of an employee leaving the company. The project aims to improve employee retention rates and reduce the costs associated with high turnover rates.

The workforce retention system can also provide actionable insights to help managers and HR professionals develop targeted retention strategies. For example, if the model identA workforce retention system using machine learning involves developing an algorithm that can predict the likelihood of an employee leaving the company. The machine learning model can be trained on historical data to identify patterns and factors that contribute to employee turnover. The model can then be used to make predictions about future turnover rates and identify the specific factors that are driving employee attrition.. The project aims to improve employee retention rates and reduce the costs associated with high turnover rates.

Overall, a workforce retention system using machine learning can help companies improve employee retention rates, reduce costs associated with turnover, and create a more engaged and satisfied workforce.

# **Technical Architecture:**



# **Project Flow:**

* User interacts with the UI to enter the input.
* Entered input is analysed by the model which is integrated.
* Once model analyses the input the prediction is showcased on the UI

To accomplish this, we have to complete all the activities listed below,

* Define Problem / Problem Understanding
  + Specify the business problem
  + Business requirements
  + Literature Survey
  + Social or Business Impact.
* Data Collection & Preparation
  + Collect the dataset
  + Data Preparation
* Exploratory Data Analysis
  + Descriptive statistical
  + Visual Analysis
* Model Building
  + Training the model in multiple algorithms
  + Testing the model
* Performance Testing & Hyperparameter Tuning
  + Testing model with multiple evaluation metrics
  + Comparing model accuracy before & after applying hyperparameter tuning
* Model Deployment
  + Save the best model
  + Integrate with Web Framework
* Project Demonstration & Documentation
  + Record explanation Video for project end to end solution
  + Project Documentation-Step by step project development procedure

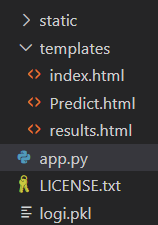
# **Prior Knowledge:**

You must have prior knowledge of the following topics to complete this project.

* ML Concepts
* Supervised learning: [https://www.javatpoint.com/supervised-machine-learning](about:blank)
* Unsupervised learning: <https://www.javatpoint.com/unsupervised-machine-learning>
* Decision tree: <https://www.javatpoint.com/machine-learning-decision-tree-classification-algorithm>
* Random forest: <https://www.javatpoint.com/machine-learning-random-forest-algorithm>
* Hyper parameter Tuning : <https://www.geeksforgeeks.org/hyperparameter-tuning/>
* Flask Basics: <https://www.youtube.com/watch?v=lj4I_CvBnt0>

**Project** **Structure:**

Create the Project folder which contains files as shown below



* We are building a flask application which needs HTML pages stored in the templates folder and a python script app.py for scripting.
* logi.pkl is our saved model. Further we will use this model for flask integration.
* Training folder contains a model training

# **Milestone 1: Data Collection & Preparation**

ML depends heavily on data. It is the most crucial aspect that makes algorithm training possible. So, this section allows you to download the required dataset.

## **Activity 1: Collect the dataset**

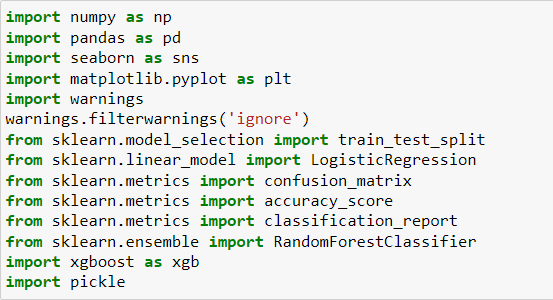
There are many popular open sources for collecting the data. Eg: kaggle.com, UCI repository, etc.

In this project we have used .csv data. This data is downloaded from kaggle.com. Please refer to the link given below to download the dataset.  
Link:<https://www.kaggle.com/datasets/gummulasrikanth/hr-employee-retention>  
As the dataset is downloaded. Let us read and understand the data properly with the help of some visualisation techniques and some analysing techniques.

Note: There are a number of techniques for understanding the data. But here we have used some of it. In an additional way, you can use multiple techniques.

## **Activity 2: Importing the libraries**

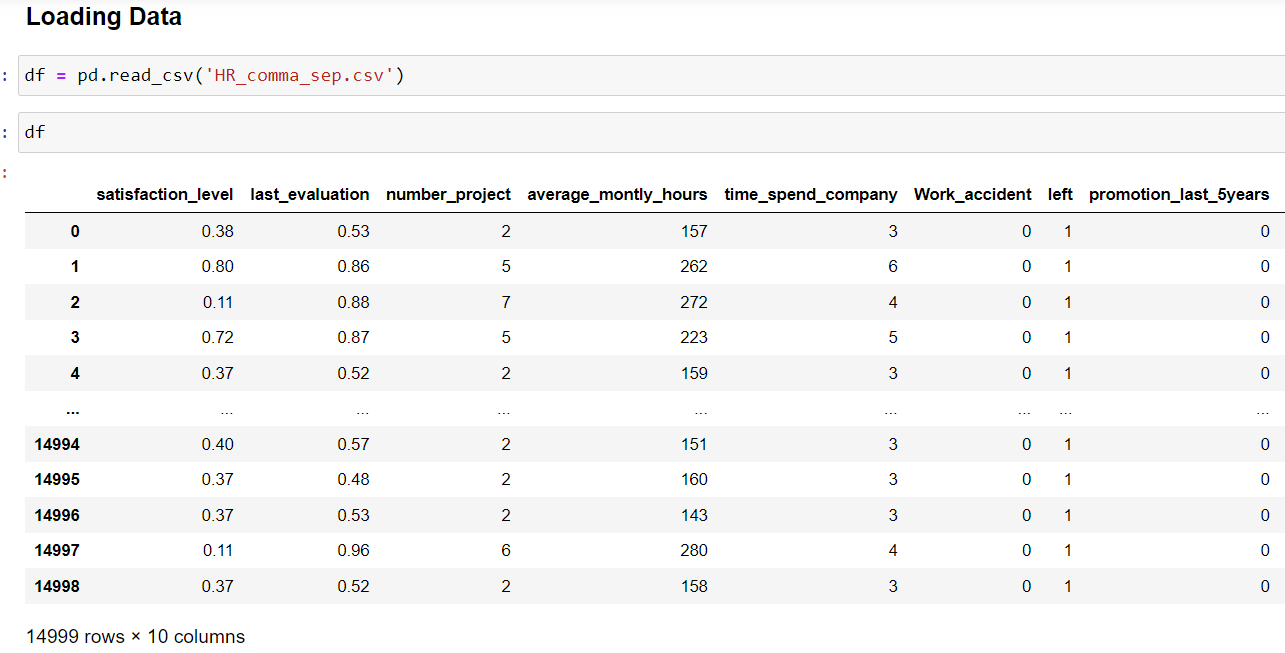
Import the necessary libraries as shown in the image. (optional) Here we have used visualisation style as fivethirtyeight.



## **Activity 3: Read the Dataset**

Our dataset format might be in .csv, excel files, .txt, .json, etc. We can read the dataset with the help of pandas.

In pandas we have a function called read\_csv() to read the dataset. As a parameter we have to give the directory of the csv file.



## **Activity 4: Data Preparation**

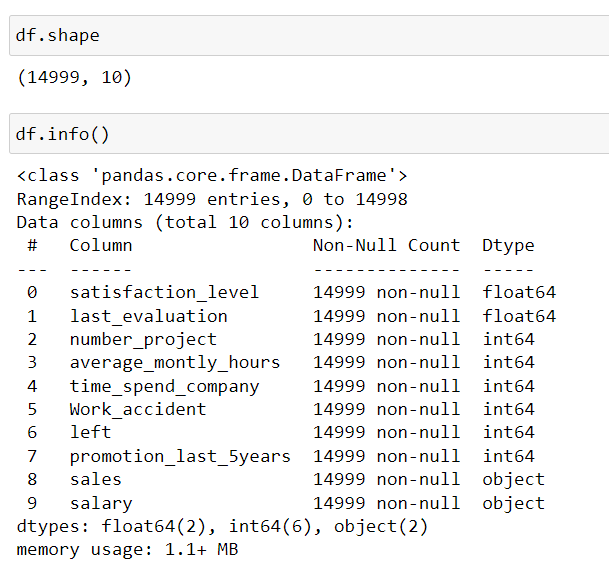
As we have understood how the data is, let's pre-process the collected data.  
The download data set is not suitable for training the machine learning model as it might have so much randomness so we need to clean the dataset properly in order to fetch good results. This activity includes the following steps.

* Handling missing values
* Handling categorical data
* Handling Outliers

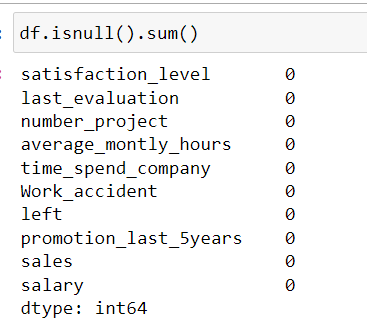
Note: These are the general steps of pre-processing the data before using it for machine learning. Depending on the condition of your dataset, you may or may not have to go through all these steps.

## **Activity 5: Handling Missing Values**

Let’s find the shape of our dataset first. To find the shape of our data, the df.shape method is used. To find the data type, df.info() function is used.

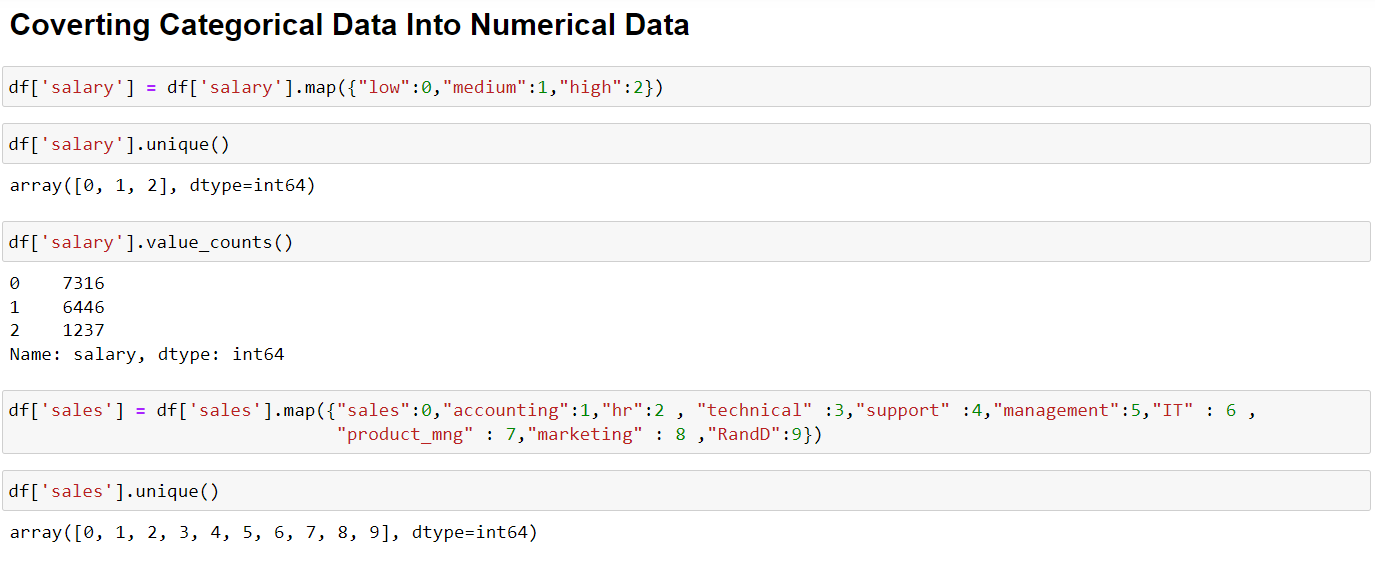


For checking the null values, df.isnull() function is used. To sum those null values we use .sum() function. From the below image we found that there are no null values present in our dataset. So we can skip handling the missing values step.



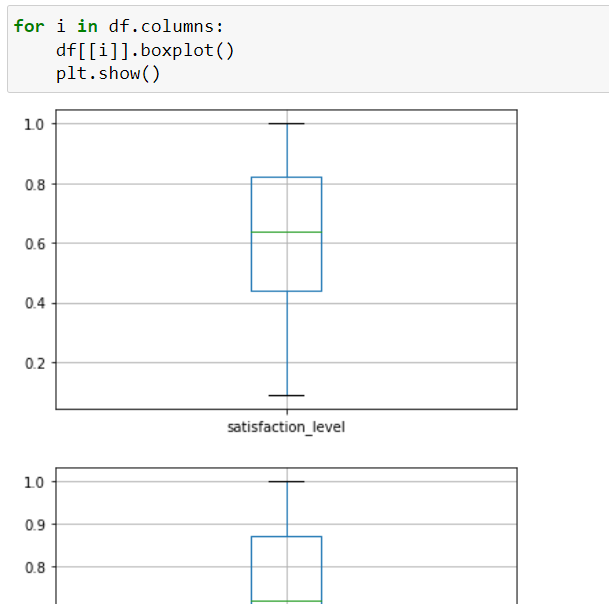
## **Activity 6: Handling Categorical Data**

For checking the Categorical data, you can use value\_counts() function. As per the given data, there is no categorical data present.



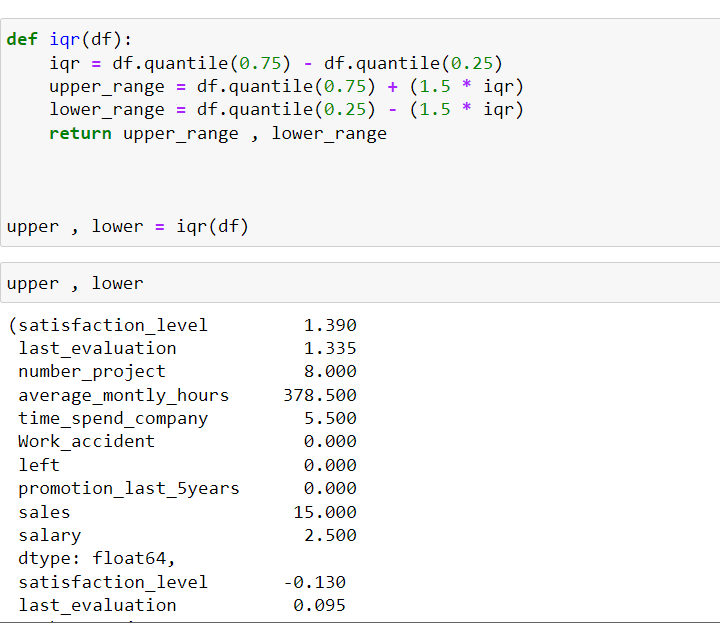
## **Activity 7: Handling outliers**

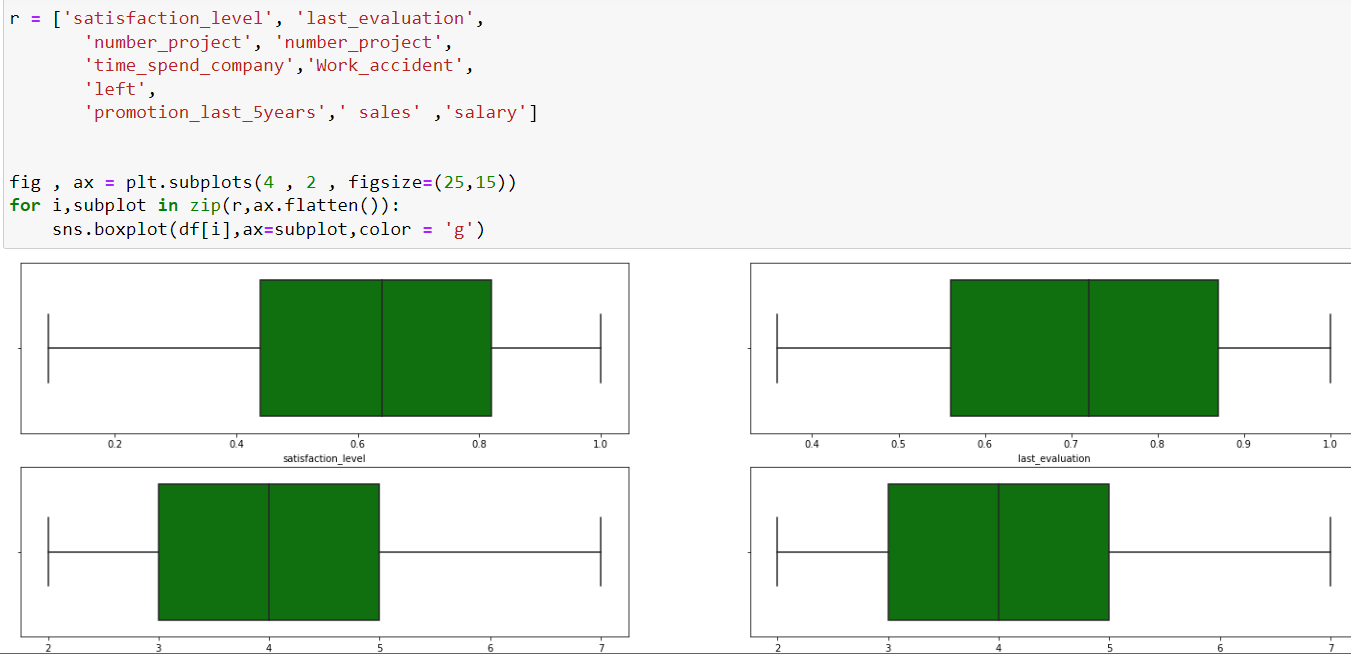
With the help of boxplot, outliers are visualized. And here we are going to find upper bound and lower bound with some mathematical formula.  
From the below diagram, we could visualize the outliers. Boxplot from seaborn library is used .



Checking with IQR and here we are going to find upper bound and lower bound with some mathematical formula. To find upper bound we have to multiply IQR (Interquartile range) with 1.5 and add it with 3rd quantile. To find lower bound instead of adding, subtract it with 1st quantile. Take image attached below as your reference.

To handle the outliers transformation technique is used and some of the outliers in the data are reduced.

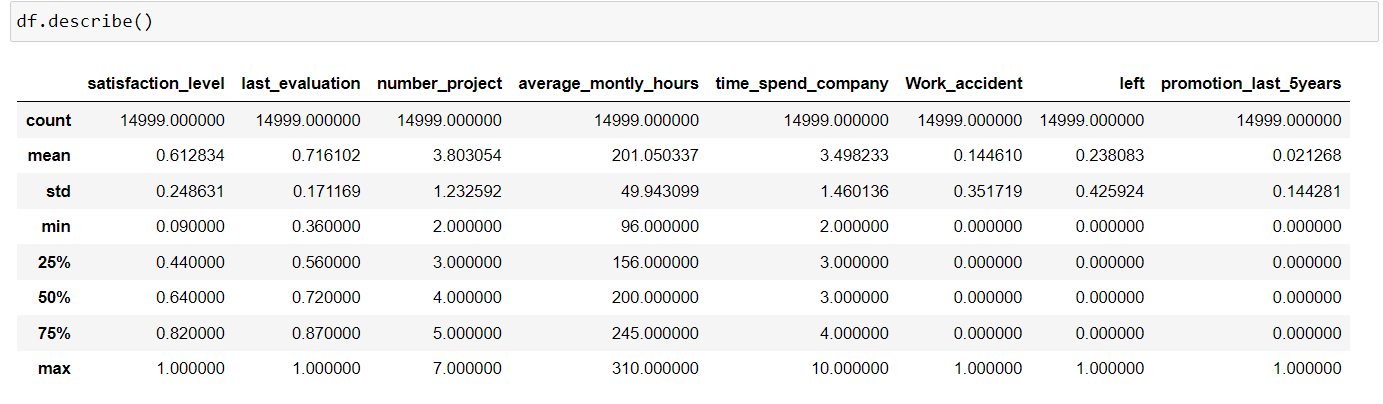




# **Milestone 2: Exploratory Data Analysis**

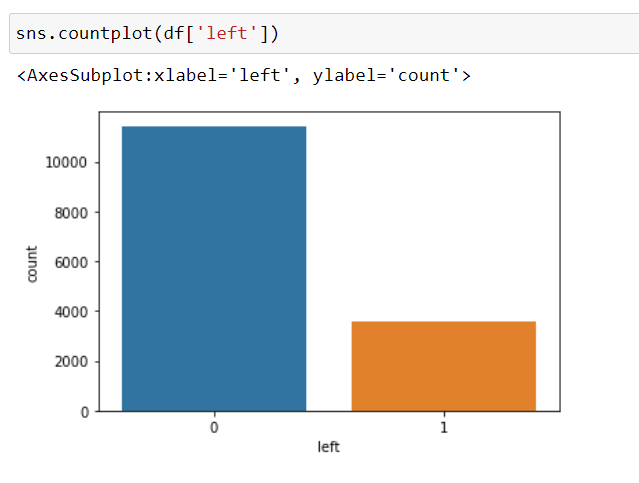
## **Activity 1: Descriptive statistics**

Descriptive analysis is to study the basic features of data with the statistical process. Here pandas has a worthy function called describe. With this describe function we can understand the unique, top and frequent values of categorical features. And we can find mean, std, min, max and percentile values of continuous features.



## **Activity 2: Visual analysis**

Visual analysis is the process of using visual representations, such as charts, plots, and graphs, to explore and understand data. It is a way to quickly identify patterns, trends, and outliers in the data, which can help to gain insights and make informed decisions.  
**Univariate analysis**  
In simple words, univariate analysis is understanding the data with single feature. Here we have displayed countplot. Seaborn package provides a wonderful function countplot.



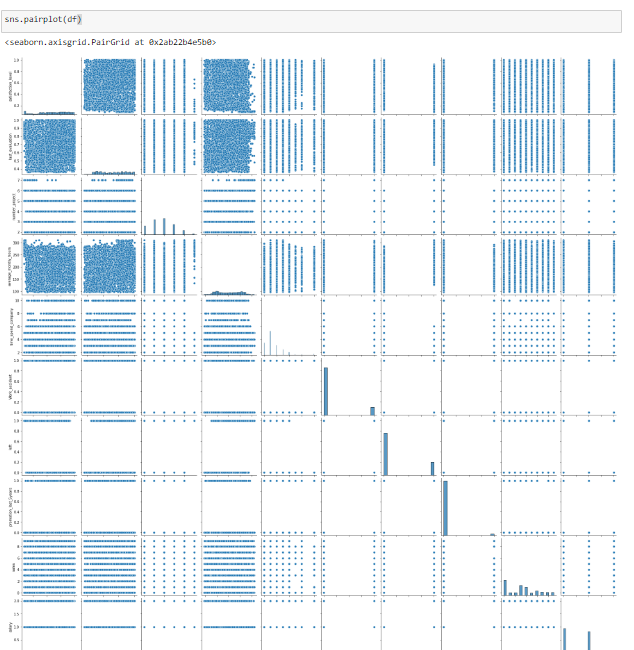
**Bivariate Analysis:**

To find the relation between two features we use bivariate analysis. You can use seaborn package to plot visualisation uisng two variables of the dataset

**Multivariate analysis**

In simple words, multivariate analysis is to find the relation between multiple features. Here we have used pairplot from seaborn package.

From the below image, we came to a conclusion that risk is predicted.

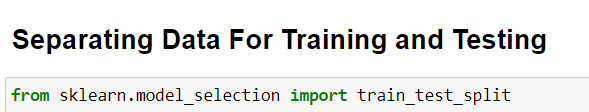


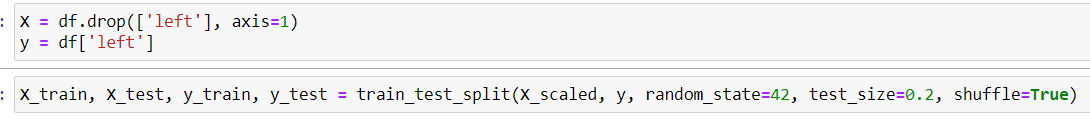
**Splitting data into train and test**

Now let’s split the Dataset into train and test sets. First split the dataset into x and y and

then split the data set

Here x and y variables are created. On x variable, df is passed with dropping the target variable. And on y target variable is passed. For splitting training and testing data we are using train\_test\_split() function from sklearn. As parameters, we are passing x, y,





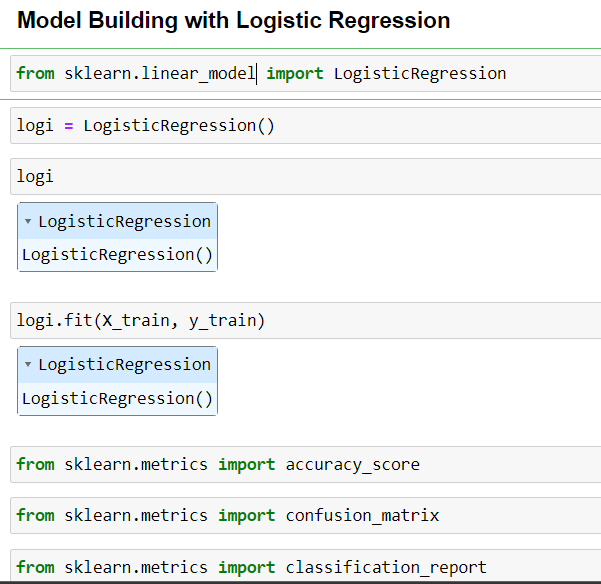
# **Milestone 3: Model Building**

## **Activity 1: Training the model in multiple algorithms**

Now our data is cleaned and it’s time to build the model. We can train our data on different algorithms. For this project we are applying three classification algorithms. The best model is saved based on its performance.

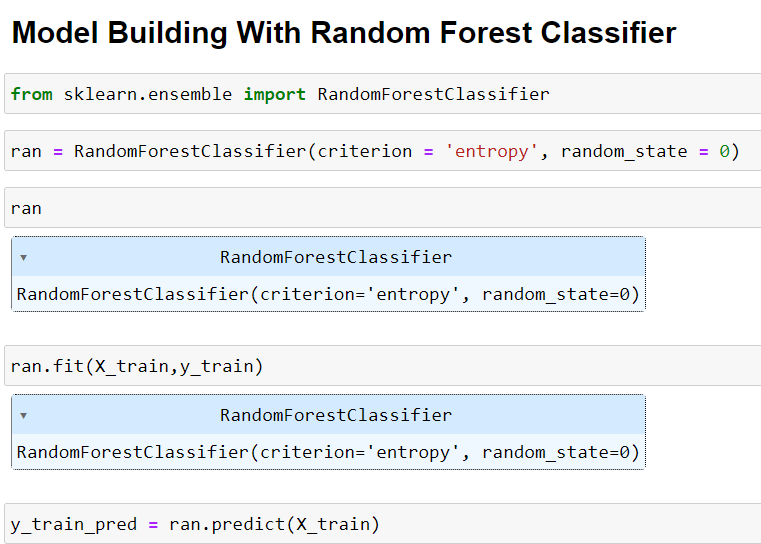
**Logistic Regression model**

A function named Logistic Regressor is created and train and test data are passed as the parameters. Inside the function, Logistic Regression model algorithm is initialised and training data is passed to the model with the .fit() function. Test data is predicted with.predict() function and saved in a new variable. For evaluating the model, a confusion matrix and classification report is done.



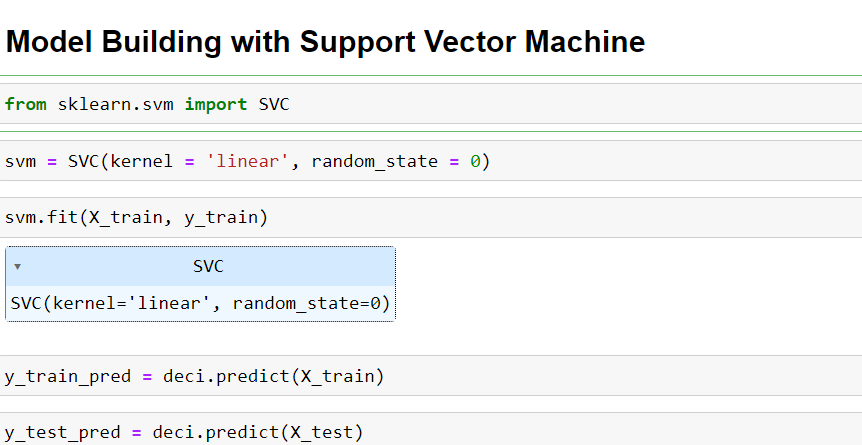
**Random Forest Classifier**

A function named Random Forest is created and train and test data are passed as the parameters. Inside the function, Random Forestalgorithm is initialised and training data is passed to the model with .fit() function. Test data is predicted with .predict() function and saved in a new variable. For evaluating the model, a confusion matrix and classification report is done.



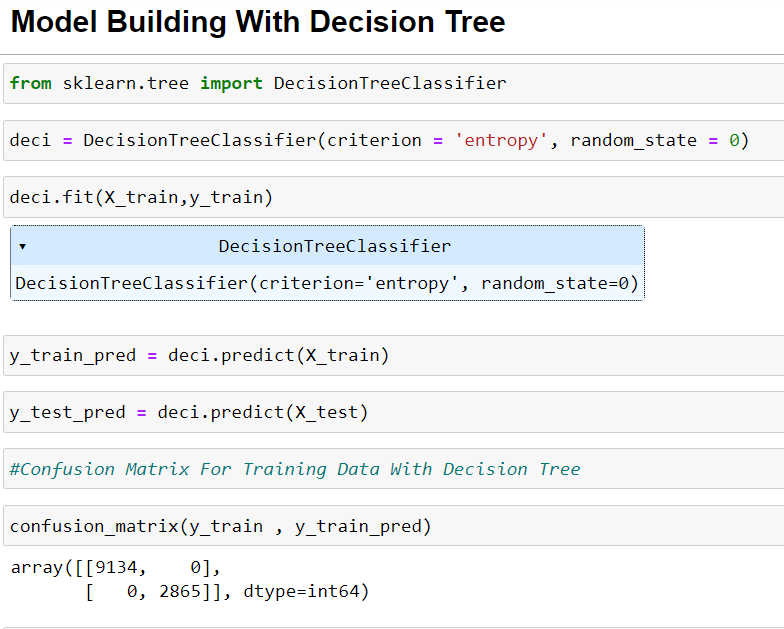
**SVM model**

A function named SVM is created and train and test data are passed as the parameters. Inside the function, Support Vector Machine algorithm is initialised and training data is passed to the model with .fit() function. Test data is predicted with .predict() function and saved in new variable. For evaluating the model, confusion matrix and classification report is done.

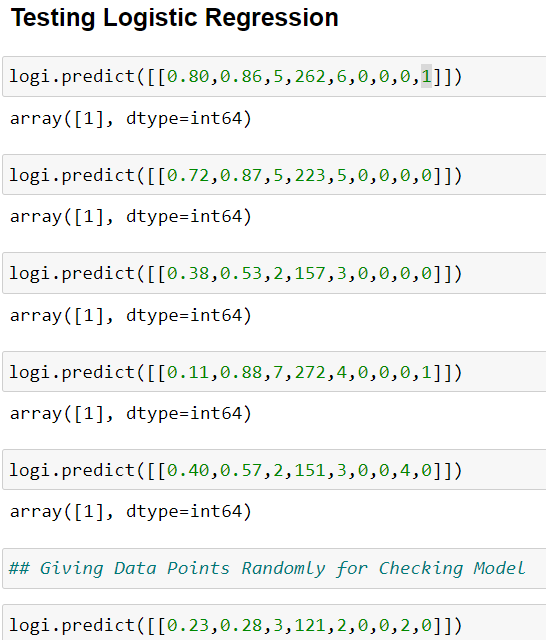
****

**Decision Tree Classifier**

 A function named Decision Tree is created and train and test data are passed as the parameters. Inside the function, Support Vector Machine algorithm is initialised and training data is passed to the model with .fit() function. Test data is predicted with .predict() function and saved in new variable. For evaluating the model, confusion matrix and classification report is done.



## **Activity 2: Testing the model**



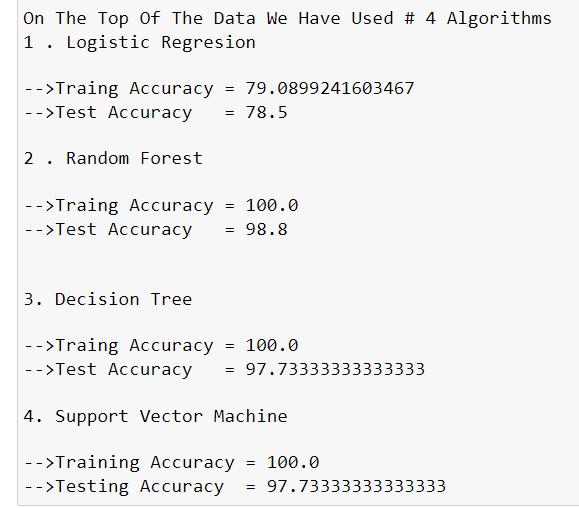
## **Activity 3: Performance testing and Hyperparameter tuning**

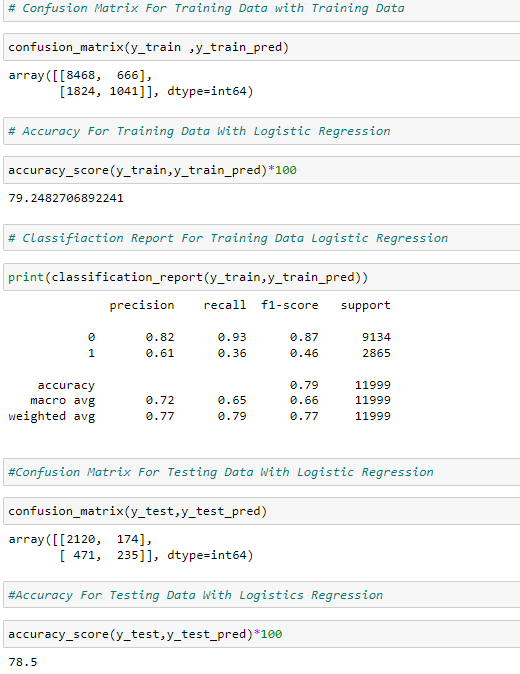
### **Testing Model With Multiple Evaluation Metrics**

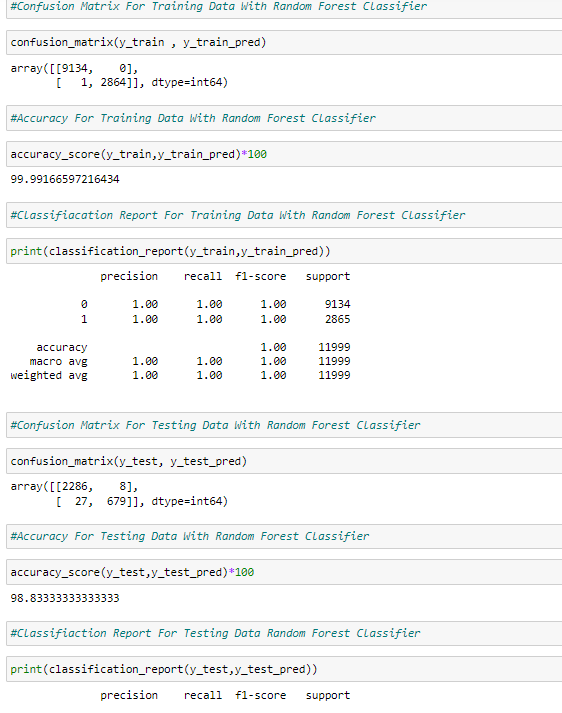
Multiple evaluation metrics means evaluating the model's performance on a test set using different performance measures. This can provide a more comprehensive understanding of the model's strengths and weaknesses. We are using evaluation metrics for classification tasks including accuracy, precision, recall, support and F1-score.

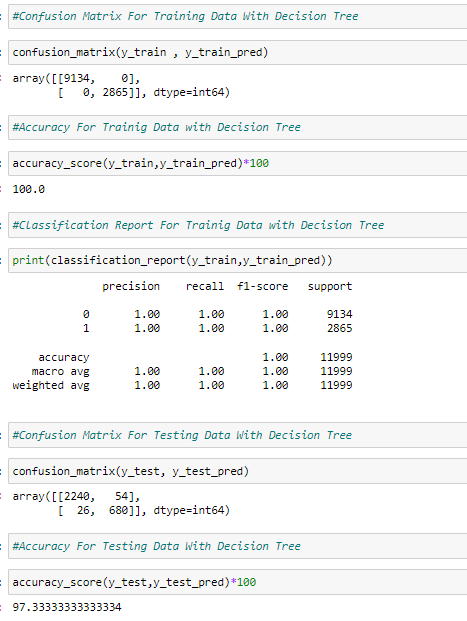
**Compare the model**

For comparing the above four models, the Comparing Model  is defined





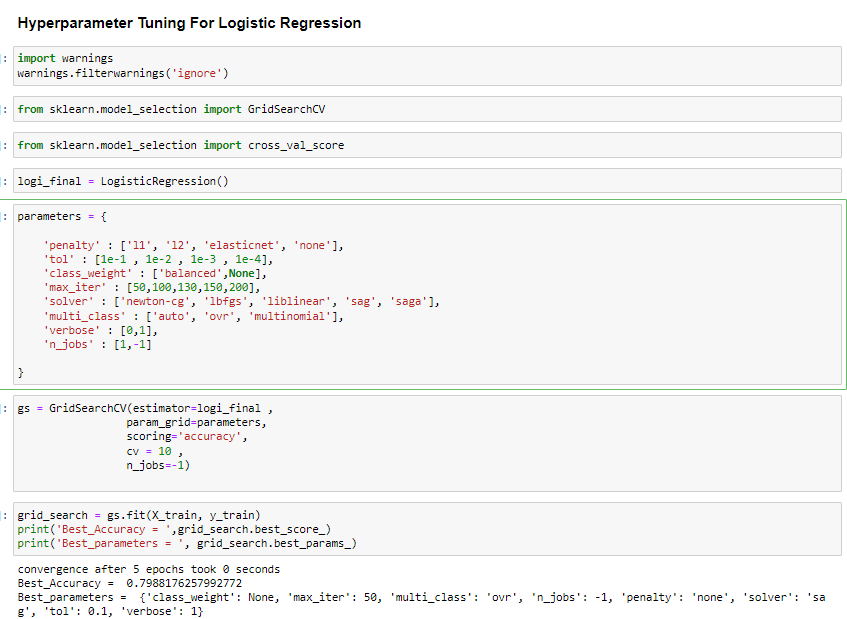




After calling the function, the results of models are displayed as output. From the four models Logistic Regression algorithm is performing well.

**Comparing model accuracy before & after applying hyperparameter tuning (Hyperparameter tuning is optional. For this project it is not required.)**

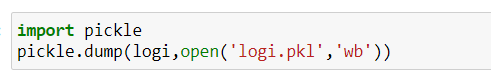
Evaluating performance of the model From sklearn, cross\_val\_score is used to evaluate the score of the model. On the parameters, we have given rf (model name), x, y, cv (as 5 folds). Our model is performing well.

****

# **Milestone 4: Model Deployment**

## **Activity 1: Save the best model**

Saving the best model after comparing its performance using different evaluation metrics means selecting the model with the highest performance and saving its weights and configuration. This can be useful in avoiding the need to retrain the model every time it is needed and also to be able to use it in the future.



## **Activity 2: Integrate with Web Framework**

In this section, we will be building a web application that is integrated to the model we built. A UI is provided for the uses where he has to enter the values for predictions. The enter values are given to the saved model and prediction is showcased on the UI.  
This section has the following tasks

* Building HTML Pages
* Building server-side script
* Run the web application

## **Activity 3: Building Html Pages**

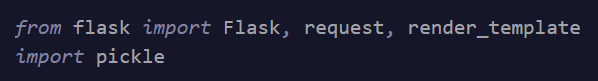
For this project create two HTML files namely

* Index.html
* Predict.html
* result.html

and save them in the templates folder.

## **Activity 4: Build Python code:**

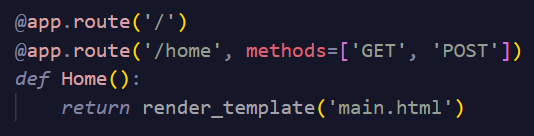
Import the libraries



Load the saved model. Importing the flask module in the project is mandatory. An object of Flask class is our WSGI application. Flask constructor takes the name of the current module ( name ) as argument.



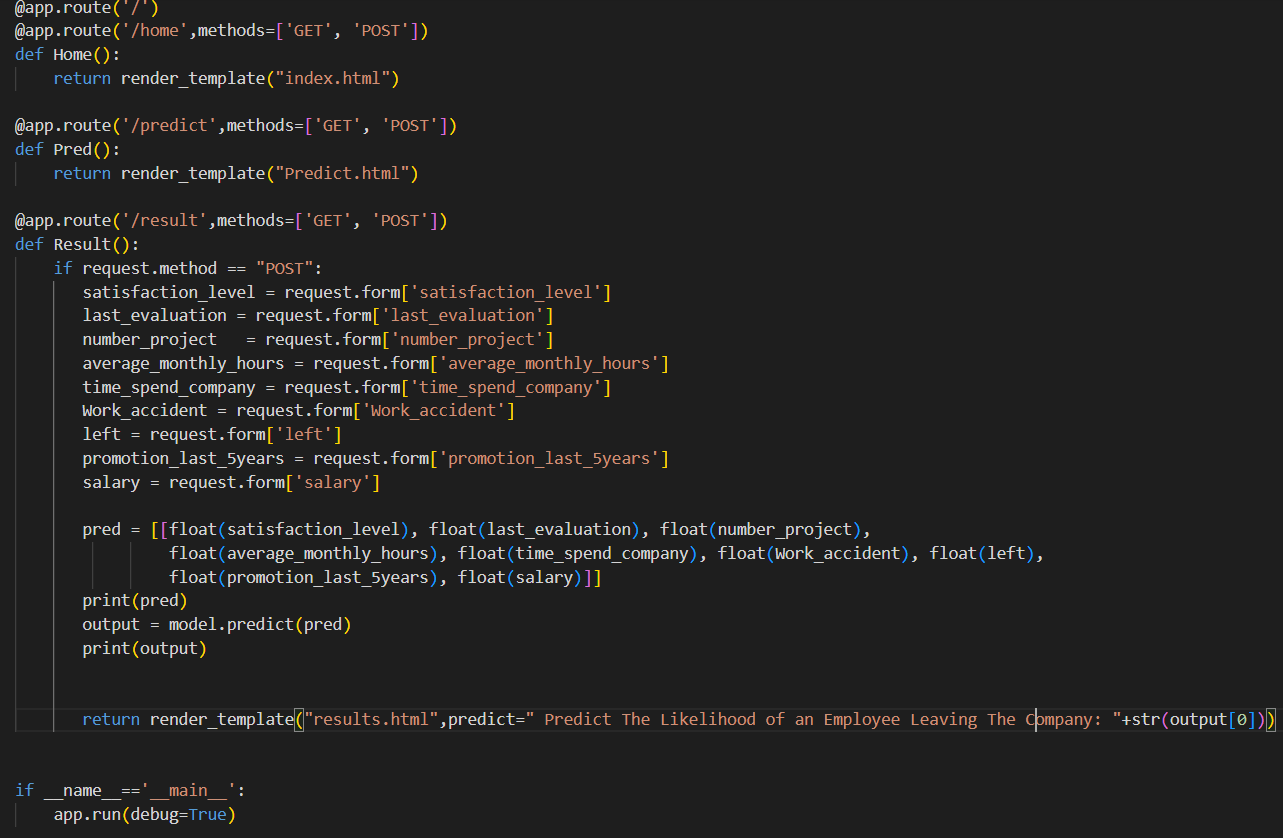
Render HTML page:



Here we will be using a declared constructor to route to the HTML page which we have created earlier.

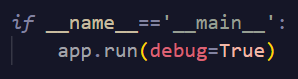
In the above example, ‘/’ URL is bound with the home.html function. Hence, when the home page of the web server is opened in the browser, the html page will be rendered. Whenever you enter the values from the html page the values can be retrieved using POST Method.

Retrieves the value from UI:



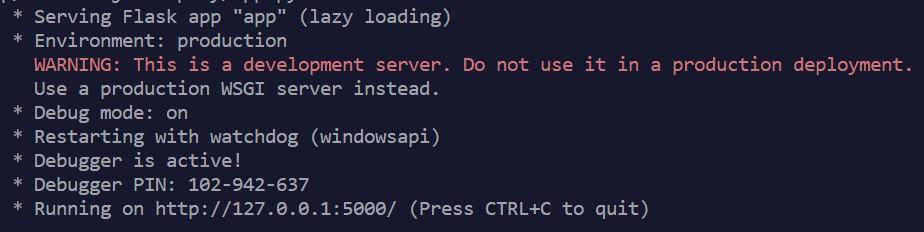
Here we are routing our app to predict() function. This function retrieves all the values from the HTML page using Post request. That is stored in an array. This array is passed to the model.predict() function. This function returns the prediction. And this prediction value will be rendered to the text that we have mentioned in the submit.html page earlier.

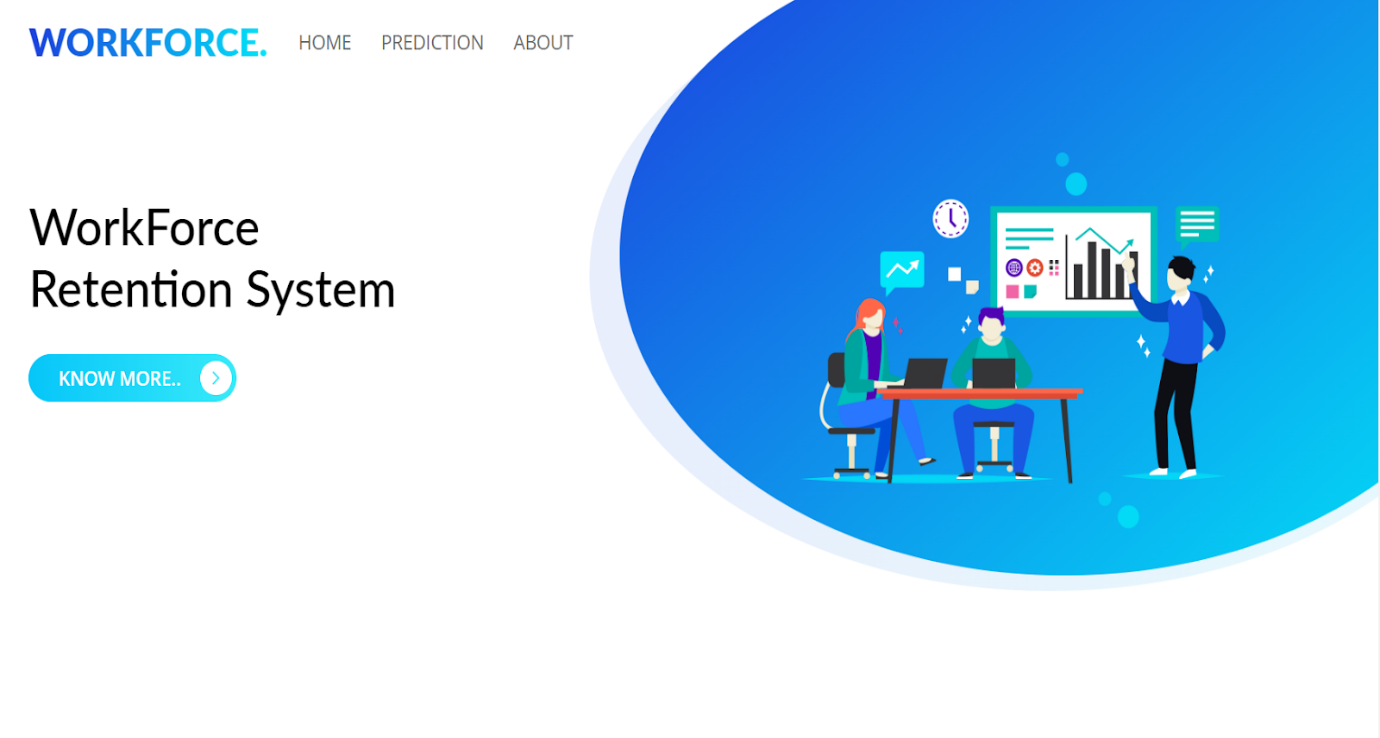
**Main Function:**



## **Activity 5: Run the web application**

* Open anaconda prompt from the start menu
* Navigate to the folder where your python script is.
* Now type “python app.py” command
* Navigate to the localhost where you can view your web page.
* Click on the predict button from the top left corner, enter the inputs, click on the submit button, and see the result/prediction on the web.



Now,Go the web browser and write the localhost url (http://127.0.0.1:5000) to get the below result:

From below picture we need to give the values of Satisfaction level, last evaluation, Number of Projects , Average Monthly Hours , Time Spend Company, Work Accident, Left, Promotion Last 5 Years, Salary,    By Giving These values the that can predict which employees are at risk of leaving an organization and develop targeted retention strategies to improve employee retention rates.

