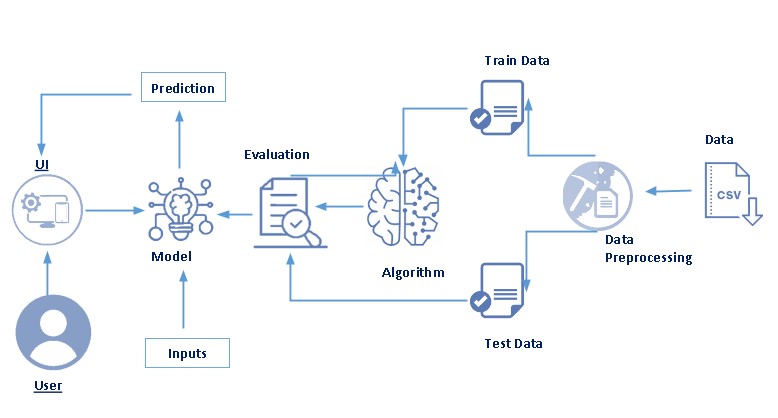
Perinatal health risk predictors using artificial intelligence

**Project Description:**

Advances in public health and medical care have enabled better pregnancy and birth outcomes. The rates of perinatal health indicators such as maternal mortality and morbidity; fetal, neonatal, and infant mortality; low birthweight; and preterm birth have reduced over time. However, they are still a public health concern, and considerable disparities exist within and between countries. For perinatal researchers who are engaged in unraveling the tangled web of causation for maternal and child health outcomes and for clinicians involved in the care of pregnant women and infants, artificial intelligence offers novel approaches to prediction modeling, diagnosis, early detection, and monitoring in perinatal health. Machine learning, a commonly used artificial intelligence method, has been used to predict preterm birth, birthweight, preeclampsia, mortality, hypertensive disorders, and postpartum depression. Real-time electronic health recording and predictive modeling using artificial intelligence have found early success in fetal monitoring and monitoring of women with gestational diabetes especially in low-resource settings. Artificial intelligence–based methodologies have the potential to improve prenatal diagnosis of birth defects and outcomes in assisted reproductive technology too.

# Technical Architecture:



# Pre requisites:

**To complete this project, you must require following software’s, concepts and packages**

* **Anaconda navigator and spider:**
  + Refer the link below to download anaconda navigator
  + Link : <https://youtu.be/1ra4zH2G4o0>
* **Python packages:**
  + Open anaconda prompt as administrator
  + Type “pip install numpy” and click enter.
  + Type “pip install pandas” and click enter.
  + Type ”pip install matplotlib” and click enter.
  + Type ”pip install pickle-mixin” and click enter.
  + Type ”pip install seaborn” and click enter.
  + Type “pip install Flask” and click enter.

# Prior Knowledge:

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

* **ML Concepts**
  + Supervised learning: <https://www.javatpoint.com/supervised-machine-learning>
  + Unsupervised learning: <https://www.javatpoint.com/unsupervised-machine-learning>
  + Classification
  + Decision tree: <https://www.javatpoint.com/machine-learning-decision-tree-classification-algorithm>
  + Random forest: <https://www.javatpoint.com/machine-learning-random-forest-algorithm>
  + KNN: <https://www.javatpoint.com/k-nearest-neighbor-algorithm-for-machine-learning>
  + Xgboost: <https://www.analyticsvidhya.com/blog/2018/09/an-end-to-end-guide-to-understand-the-math-behind-xgboost/>
  + Evaluation metrics: <https://www.analyticsvidhya.com/blog/2019/08/11-important-model-evaluation-error-metrics/>
* **Flask Basics** : <https://www.youtube.com/watch?v=lj4I_CvBnt0>

# Project Objectives:

By the end of this project you will:

* Know fundamental concepts and techniques used for machine learning.
* Gain a broad understanding about data.
* Have knowledge on pre-processing the data/transformation techniques on outlier and some visualization concepts.

# Project Flow:

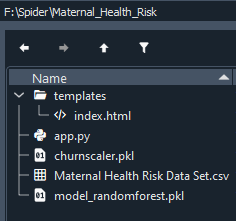
* User interacts with the UI to enter the input.
* Entered input is analyzed by the model which is integrated.
* Once model analyses the input the prediction is showcased on the UI

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

* Data collection
  + Collect the dataset or create the dataset
* Visualizing and analyzing data
  + Univariate analysis
  + Bivariate analysis
  + Multivariate analysis
  + Descriptive analysis
* Data pre-processing
  + Checking for null values
  + Handling outlier
  + Handling categorical data
  + Splitting data into train and test
* Model building
  + Import the model building libraries
  + Initializing the model
  + Training and testing the model
  + Evaluating performance of model
  + Save the model
* Application Building
  + Create an HTML file
  + Build python code

# 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.
* Model.pkl is our saved model. Further we will use this model for flask integration.
* Training folder contains model training files and training\_ibm folder contains IBM deployment files.

# Milestone 1: Data Collection

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

**Activity 1: Download the dataset**

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

In this project we have used Maternal Health Risk Data Set.csv data. This data is downloaded from UC Irvine Machine Learning Repository. Please refer the link given below to download the dataset.

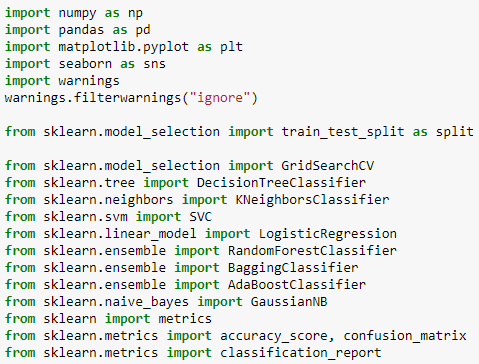
Link: <https://archive.ics.uci.edu/ml/datasets/Maternal+Health+Risk+Data+Set>

# Milestone 2: Visualizing and analysing the data

As the dataset is downloaded. Let us read and understand the data properly with the help of some visualization techniques and some analysing techniques.

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

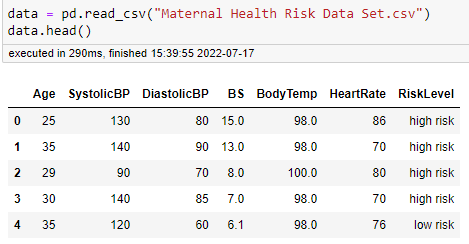
**Activity 1: Importing the libraries**

**Import the necessary libraries as shown in the image.** 

**Activity 2: 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.

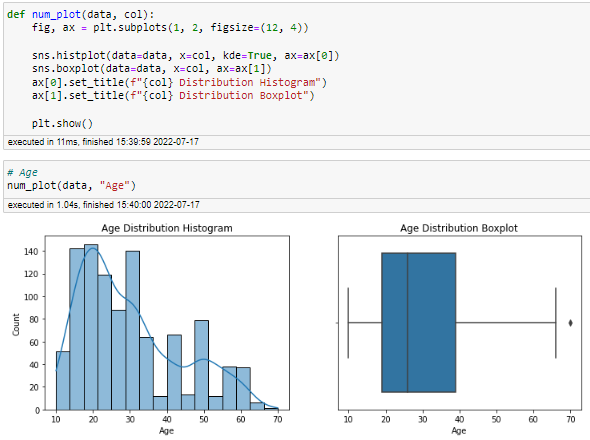
In pandas we have a function called read\_csv() to read the dataset. As a parameter we have to give the directory of csv file.



**Activity 3: Univariate analysis**

In simple words, univariate analysis is understanding the data with single feature. Here we have displayed two different graphs such as distplot and countplot.

* Seaborn package provides a wonderful function histplot.and boxplot With the help of histplot, and boxplot we can find the distribution of the feature. To make multiple graphs in a single plot, we use subplot.



**Activity 4: Bivariate analysis**

To find the relation between two features we use bivariate analysis. Here we are visualizing the relationship between drug & BP, drug & sex and drug & cholesterol.

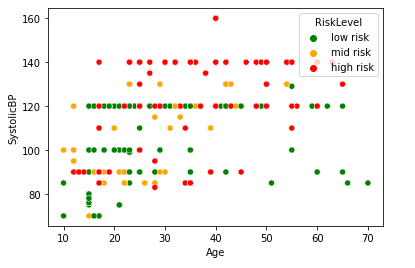
* Histplot is used here. As a 1st parameter we are passing x value and as a 2nd parameter we are passing hue value.
* From the below plot you can understand that as Age increases the patients Risk level increases.



With the help of age feature we are creating an age interval and finding the relation between RiskLevel feature and age interval feature. Function histplot is used to find the relationship. From the above image we get a clear understanding, RiskLevel increases only for patients above age 30 years.

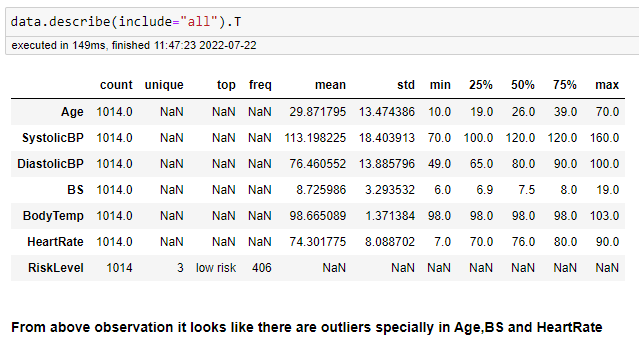
**Activity 5: Multivariate analysis**

In simple words, multivariate analysis is to find the relation between multiple features. Here we have used scatterplot from seaborn package. From the below image, we came to a conclusion Pregnant women with high SystolicBP seems to have a high health risk, regardless of their age.



**Activity 6: Descriptive analysis**

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.



# Milestone 3: Data Pre-processing

As we have understood how the data is lets pre-process the collected data.

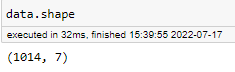
The download data set is not suitable for training the machine learning model as it might have so much of randomness so we need to clean the dataset properly in order to fetch good results. This activity includes the following steps.

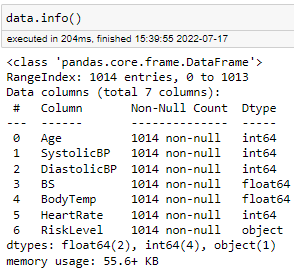
* Handling missing values
* Handling categorical data
* Handling outliers
* Scaling Techniques
* Splitting dataset into training and test set

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.

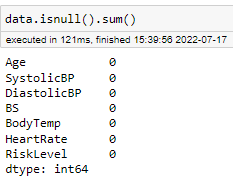
**Activity 1: Checking for null values**

* Let’s find the shape of our dataset first, To find the shape of our data, 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 to it. From the below image we found that there are no null values present in our dataset. So we can skip handling of missing values step.

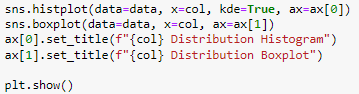


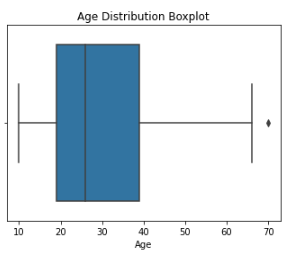
Let’s look for any outliers in the dataset

**Activity 2: Handling outliers**

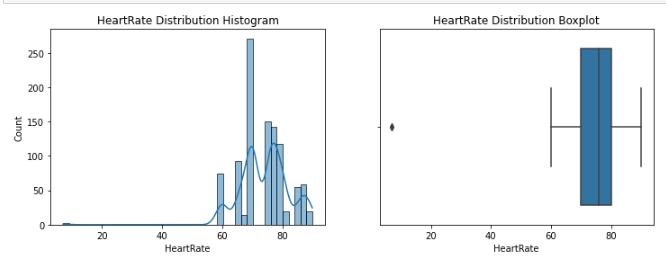
With the help of boxplot, outliers are visualized. And here we are going to find upper bound and lower bound of all feature with some mathematical formula.

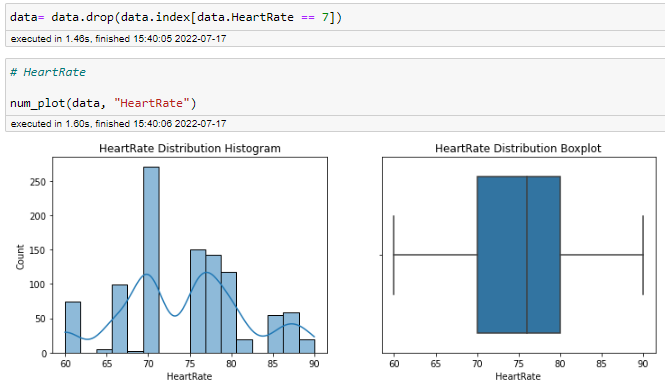
* From the below diagram, we could visualize that age feature has outliers. Boxplot from seaborn library is used here.





* After looking for outliers in all features we see that most of the data is practically possible other than HeartRate. The row containing the outlier is removed.





**Activity 3: Handling Categorical Values**

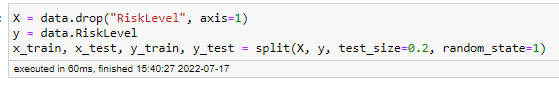
Since in our data the only categorical feature is dependent feature and all independent features are numerical handling he categorical values is not necessary in this model.

**Activity 4: Splitting data into train and test**

Now let’s split the Dataset into train and test sets

Changes: 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, test\_size, random\_state.



# Milestone 4: Model Building

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 eight classification algorithms. The best model is saved based on its performance.

**1: Decision tree model**

**2: KNN model**

**3: SVC model**

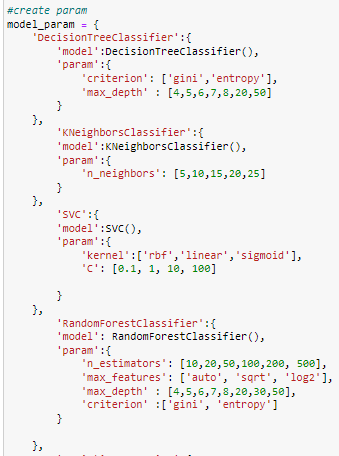
**4: Random forest model**

**5: LogisticRegression**

**6: BaggingClassifier**

**7: AdaBoostClassifier**

**8: Naive bayes**







Now let’s see the performance of all the models and save the best model

**Compare the model**

For comparing the above eight models GridSearchCV function is defined.



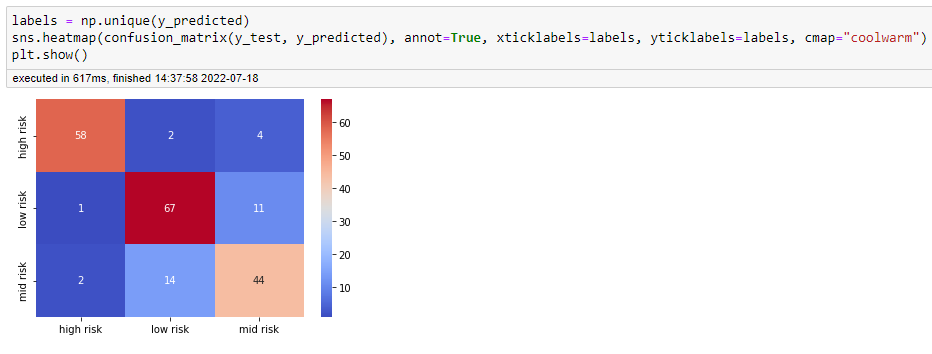
After calling the function, the results of models are displayed as output. From the eight model random forest and is performing well. From the below image, We can see the accuracy of the model. Models have 82.9% accuracy. Even confusion matrix also have same results. Training time of decision tree is faster than random forest. In such case we have to select decision tree model (time saving & cost wise profitable). But, here random forest is selected and evaluated with cross validation. Additionally, we can tune the model with hyper parameter tuning techniques.

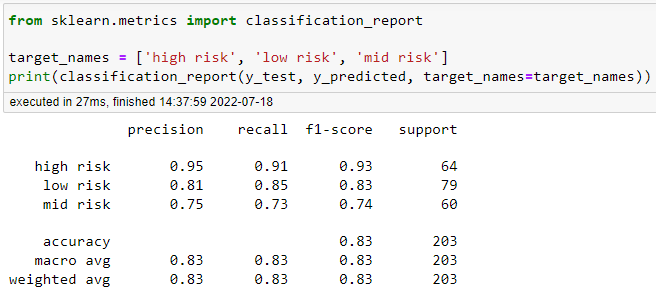
**Evaluating performance of the model and saving the model**

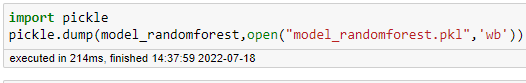
From sklearn.metrics: confusion\_matrix and classification\_report is used to evaluate the score of the model. On the parameters, we have given y\_test,y\_predicted for confusion\_matrix and y\_test, y\_predicted, target\_names=target\_names for classification\_report.

Our model is performing well. So, we are saving the model by pickle.dump().

.







# Milestone 5: Application Building

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 serverside script

**Activity1: Building Html Pages:**

For this project create three HTML files namely

* home.html
* predict.html
* submit.html

and save them in templates folder.

Let’s see how our about.html page looks like:



Now when you click on predict button you will get redirected to predict.html

Lets look how our index.html file looks like:



Now when you click on Predict button from left bottom corner you will get redirected to /y\_predict

Lets look how our submit.html file looks like:



**Activity 2: Build Python code:**

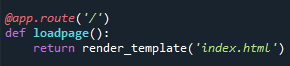
Import the libraries



Load the saved model. Importing 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.



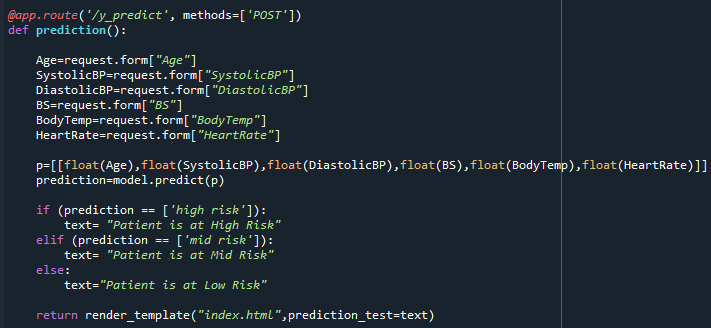
Render HTML page:



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

In the above example, ‘/’ URL is bound with home.html function. Hence, when the home page of the web server is opened in 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 y\_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 rendered to the text that we have mentioned in the submit.html page earlier.

Main Function:



**Activity 3: Run the 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 right corner, enter the inputs, click on the submit button, and see the result/prediction on the web.

