MACHINE LEARNING MODEL TO PREDICT MATERNAL HEALTH RISK LEVEL

INURI MUTHUKUMARANA

30TH APRIL 2022

INTRODUCTION

- Maternal mortality is unacceptably high, especially in rural areas due to lack of information and quality health services. Every day, approximately 810 women die from preventable causes related to pregnancy and childbirth (WHO).
- Early identification of potential health risk allows physicians to come up with comprehensive intervention strategies to minimize maternal deaths and pregnancy-related complications.
- However, more human effort and time is required to manually analyze the health data of pregnant women to identify risk intensity levels.
- Therefore, a machine learning model would be of great benefit for early detection and management of risk factors associated with pregnancy.

DATA

UCI

Machine Learning Repository

Center for Machine Learning and Intelligent Systems

Check out the beta version of the new UCI Machine Learning Repository we are currently testing! Contact us if you have a

Maternal Health Risk Data Set Data Set

Download: Data Folder, Data Set Description

Abstract: Data has been collected from different hospitals, community clinics, maternal health cares from the rural areas of Bangladesh

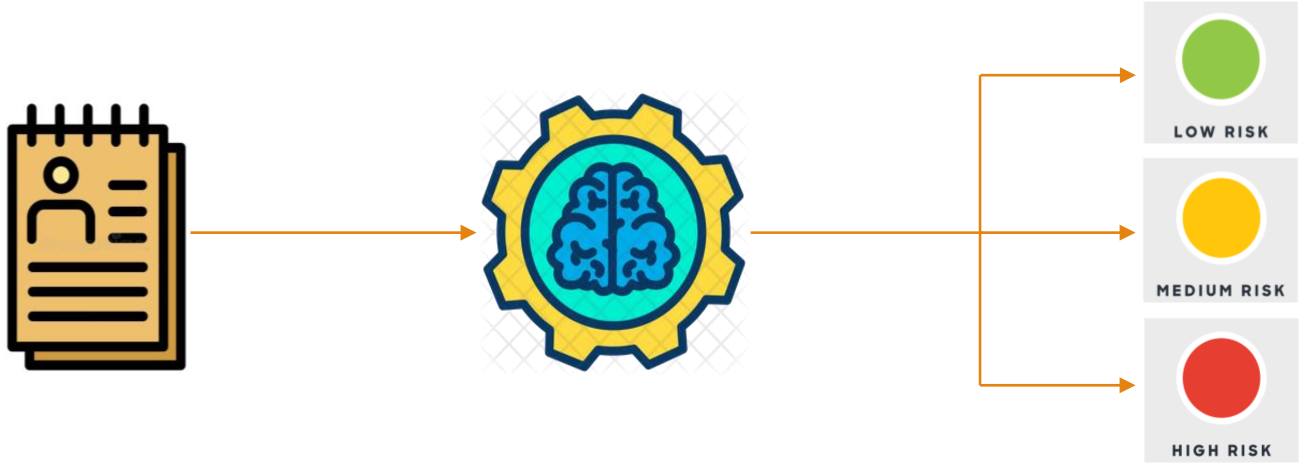
Data Set Characteristics:	N/A	**Number of Instances:**	1014	**Area:**	Life
Attribute Characteristics:	N/A	**Number of Attributes:**	7	**Date Donated**	2020-12-31

Age	SystolicBP	DiastolicBP	BS	BodyTemp	HeartRate	RiskLevel
25	130	80	15.0	98.0	86	high risk
35	140	90	13.0	98.0	70	high risk
29	90	70	8.0	100.0	80	high risk
30	140	85	7.0	98.0	70	high risk
35	120	60	6.1	98.0	76	low risk

- Data Source https://archive.ics.uci.edu/ml/datasets/Maternal+Health+Risk+Data+Set
- File Format CSV file
- File size 30 KB (7 Columns and 1014 rows)
- Data Attributes
  Age in years
  Systolic blood pressure in mmHg
  Diastolic blood pressure in mmHg
  Blood sugar in mmol/L
  Body temperature in F
  Heart rate in beats per minute
  Risk intensity level (Low, Mid and High)

SOLUTION APPROACH & TOOLS

- Multiclass Classification Problem



INPUT
Health Data
Age, SystolicBP, DiastolicBP, BS, BodyTemp, HeartRate

ML Model
Multiclass Classifier

OUTPUT
Risk Level
Low(0), Mid(1), High(2)

- Tool Box

python™
NumPy
pandas
matplotlib
scikit learn
seaborn

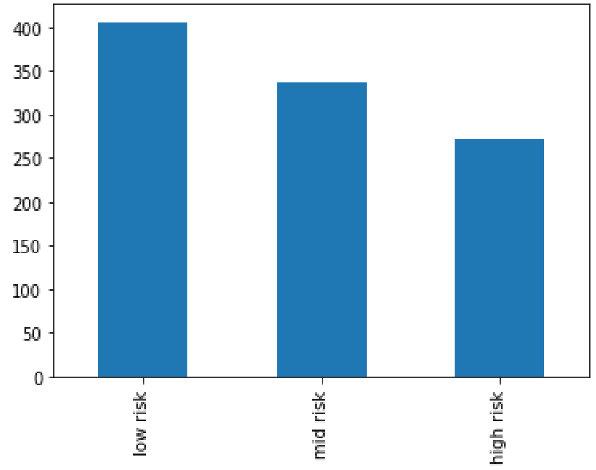
METHODOLOGY

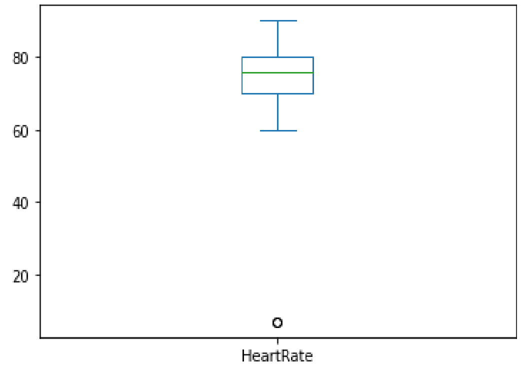
Pre-processing, Exploratory Data Analysis

- **Check data types**
- **Descriptive statistics**
- **Graphs (Bar charts, Boxplots)**
- **Treat missing values**
- **Treat Outliers**
- **Encoding**

```
Column       Non-Null Count  Dtype
------       --------------  -----
Age          1014 non-null   int64
SystolicBP   1014 non-null   int64
DiastolicBP  1014 non-null   int64
BS           1014 non-null   float64
BodyTemp     1014 non-null   float64
HeartRate    1014 non-null   int64
RiskLevel    1014 non-null   object
```

	count	unique	top	freq	mean	std	min	25%	50%	75%	max
Age	1014.0	NaN	NaN	NaN	29.871795	13.474386	10.0	19.0	26.0	39.0	70.0
SystolicBP	1014.0	NaN	NaN	NaN	113.198225	18.403913	70.0	100.0	120.0	120.0	160.0
DiastolicBP	1014.0	NaN	NaN	NaN	76.460552	13.885796	49.0	65.0	80.0	90.0	100.0
BS	1014.0	NaN	NaN	NaN	8.725986	3.293532	6.0	6.9	7.5	8.0	19.0
BodyTemp	1014.0	NaN	NaN	NaN	98.665089	1.371384	98.0	98.0	98.0	98.0	103.0
HeartRate	1014.0	NaN	NaN	NaN	74.301775	8.088702	7.0	70.0	76.0	80.0	90.0
RiskLevel	1014	3	low risk	406	NaN	NaN	NaN	NaN	NaN	NaN	NaN






```python
# Assign numbers to class labels
data['y_act'] = data['RiskLevel']
data['y_act'].replace('low risk', 0, inplace=True)
data['y_act'].replace('mid risk', 1, inplace=True)
data['y_act'].replace('high risk', 2, inplace=True)
```

```python
#Calculate Upper & Lower bound
q3 = np.quantile(data['HeartRate'], 0.75)
q1 = np.quantile(data['HeartRate'], 0.25)
IQR = q3-q1
upper_limit = q3 + 1.5 * IQR
lower_limit = q1 - 1.5 * IQR
data.loc[(data['HeartRate']<= lower_limit) | (data['HeartRate']>= upper_limit)]
```

METHODOLOGY

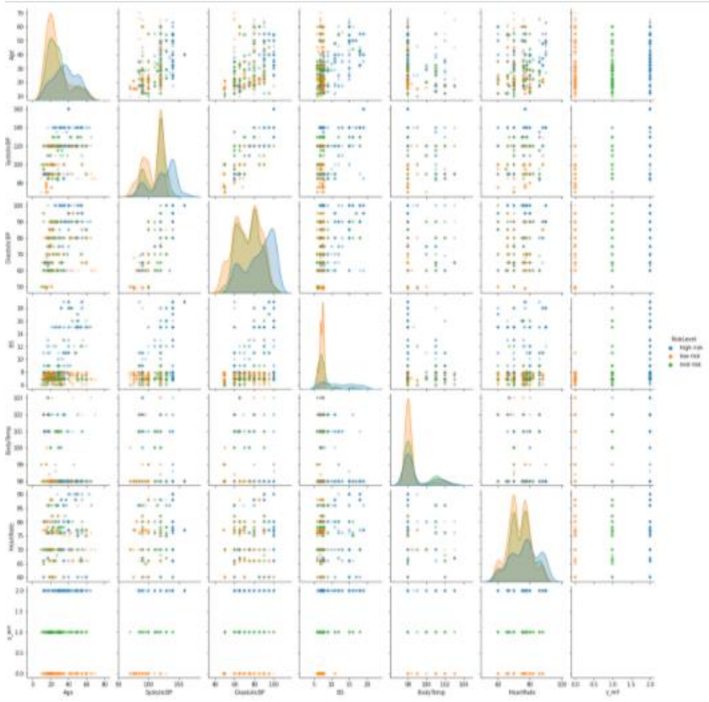
Model Building

Correlation Matrix and Heatmap

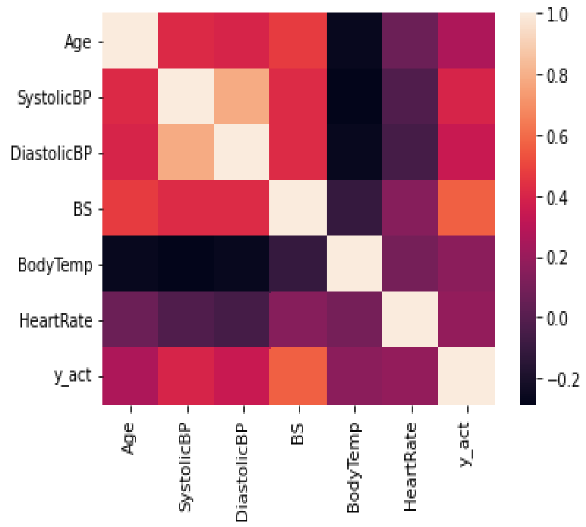
Prepare X variables and y variable

Train Test Split

Train Models



Independent variables	Age, SystolicBP, DiastolicBP, BS, BodyTemp, HeartRate
Dependent variables	y_act




```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

print(F"Train sample size = {len(X_train)}")
print(F"Test sample size  = {len(X_test)}")
```

```
Train sample size = 809
Test sample size  = 203
```

Decision Trees	**Random Forest (RF)**	**Support Vector Machines**	**k-Nearest Neighbors**

METHODOLOGY

Model Comparison

Test Models

Comparison

Hyperparameter Tuning

Randomized Search

Based on **Accuracy**, **Precision, recall** and **F1 score**, Random Forest classifier was selected

Model	accuracy	precision	recall	f1_score
Decision Tree	0.753695	0.776414	0.753695	0.752239
Random Forest	0.832512	0.838724	0.832512	0.832419
SVM	0.650246	0.635250	0.650246	0.626198
K-Neighbour	0.689655	0.689837	0.689655	0.688362

```
# Define Hyperparameter Grid
param_grid = {
              'n_estimators': [100, 200, 250],
              "max_depth": [10, 20, 50],
              "min_samples_leaf": [1, 2, 5],
              "min_samples_split": [2, 5, 10]
             }

# Create model object
model = RandomForestClassifier()

# Create RandomizedSearchCV object
model_cv = RandomizedSearchCV(estimator = model, param_distributions = param_grid, cv = 5)

model_cv.fit (X_train, y_train)

# Print the tuned parameters and score
print("Tuned Model Parameters: {}".format(model_cv.best_params_))
print("Best model score: {}".format(model_cv.best_score_))
```

```
Tuned Model Parameters: {'n_estimators': 250, 'min_samples_split': 2, 'min_samples_leaf': 1, 'max_depth': 10}
Best model score: 0.8083965953531171
```

METHODOLOGY

Model Evaluation

Evaluation Matrices

Feature Importance

Save Model

Accuracy, Precision, recall , **F1 score**, **Weighted AUC score for one vs one and one vs rest** were used as evaluation matrices.

```
# Model evaluation matrices
print("f1_score:",metrics.f1_score(test_result['y_act'], test_result['y_pred'], average='weighted'))
print('\n')

print("classification_report:\n",classification_report(test_result['y_act'], test_result['y_pred']))
```

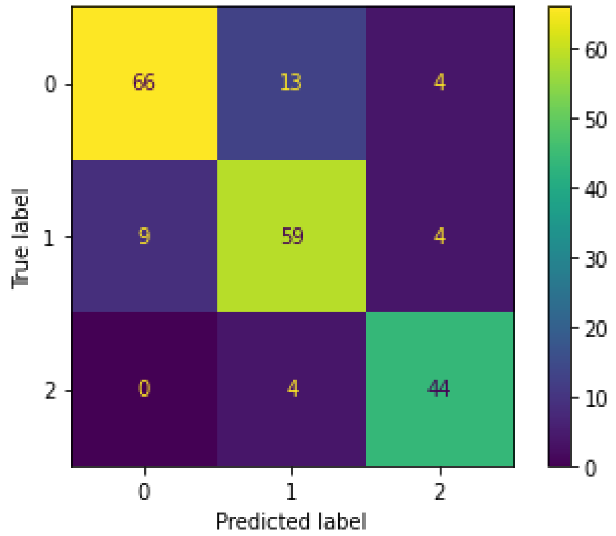
```
# Calculate AUC OVO & OVR
print("Weighted_auc_score_ovo:", metrics.roc_auc_score(y_test,model.predict_proba(X_test), multi_class="ovo", average="weighted"))
print("Weighted_auc_score_ovr:", metrics.roc_auc_score(y_test,model.predict_proba(X_test), multi_class="ovr", average="weighted"))
```

Final model was saved as pickle file

```
import pickle

save_file = 'model_rf.pickle'
pickle.dump(model, open(save_file, 'wb'))
```

RESULTS & CONCLUSION



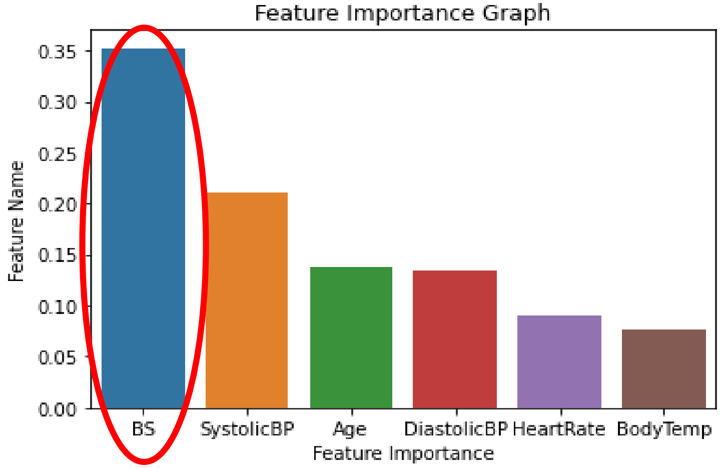

```
f1_score: 0.8324491505285918

classification_report:
              precision    recall  f1-score   support

           0       0.88      0.80      0.84        83
           1       0.78      0.82      0.80        72
           2       0.85      0.92      0.88        48

    accuracy                           0.83       203
   macro avg       0.83      0.84      0.84       203
weighted avg       0.84      0.83      0.83       203

Weighted_auc_score_ovo: 0.9291768881766804
Weighted_auc_score_ovr: 0.9212631143534011
```



Blood sugar level is identified as most significant risk factor that can increase the maternal mortality risk.

Accuracy	**0.83**
Precision (Weighted Avg)	**0.84**
Recall (Weighted Avg)	**0.83**
F1-score (Weighted Avg)	**0.83**
AUC (Weighted Avg) - OVR	**0.92**

Physicians can use this machine learning model as a supporting tool to accurately determine the **maternal health risk level** and identify of **potential risk factors.**

THANK YOU !