Prediction of patient survival from heart failure using Machine Learning and Deep Learning Algorithms

IBM Advanced Data Science – Capstone Project

Zaw Min Thant, Ph.D.

Outline of Presentation

Presentation for Stakeholders

- Introduction
  - Business issue
  - Use case
- Objectives of the study
- Methodology
- Results

Presentation for Data Science Peers

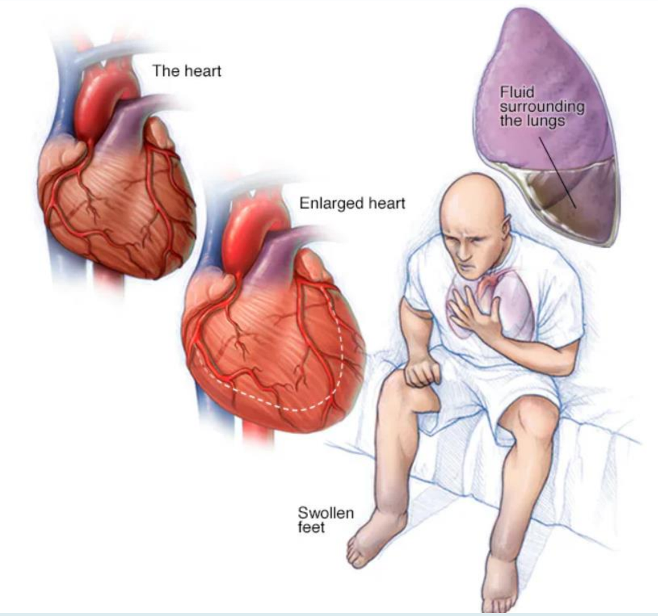
- Setting up the environment
- Extract, Transform, and Load (ELT)
- Exploratory Data Analysis (EDA)
  - Data Exploration
  - Data Visualization
- Feature Engineering
- Model algorithms (model definition & training)
- Model performance indicators
- Model evaluation
- Model tuning
- Random Forest algorithm
- References

Presentation for Stakeholders

Introduction

Business issues

- Heart failure - a life-threatening issue and has higher negative impacts
- If one encounters heart failure, the heart cannot support vital blood to the body.
- A clear understanding of which factors cause heart failure will enhance the survival chance of patients in the future.



https://www.mayoclinic.org/diseases-conditions/heart-failure/symptoms-causes/syc-20373142#dialogId65548891

Use case

- Data from **UC Irvine Machine Learning Repository**
- It was derived from *Chicco and Jurman (2020)*
- Includes cases of 299 patients, and it was collected in 2015

Objectives

- to explore which machine learning algorithms are better suited to predict the event of deceased from heart failure
- to predict the probability of deceased from heart failure and which factors most affect on heart failure

Methodology

- Apache Spark
- The target variable (DEATH_EVENT) includes a binary classification (Survived:**0** & Deceased:**1**)
- ***Logistic Regression, Random Forest, Gradient-Boosted Tree, and Artificial Neural Network***

Results

Comparison of model performance for three Machine Learning and one Deep Learning algorithms

Model	Algorithm	Evaluation		Remark
		Training accuracy	**Test accuracy**	
Logistic Regression	Machine learning	0.86	0.79	
Random Forest	Machine learning	**0.95**	**0.85**	The best fit algorithm
Gradient-Boosted Tree (GBT)	Machine learning	0.98	0.84	
Feed Forward Neural Network	Deep learning	0.87	0.79	

IBM Watson Studio Notebook permalinks GIST

https://dataplatform.cloud.ibm.com/analytics/notebooks/v2/a4eedece-d914-4691-872b-8cc8c3f1cc08/view?access_token=09fa7105a248a3d7baf512bf4ad3a24578caf45a610e8116ae664a1cebb7dc71&context=cpdaas

Presentation for Data Science Peers

Architectural Choices

Component	Technology
Development platform	Apache Spark
	IBM Watson Studio
	PySpark 3.3
	Python 3.10 Jupyter Notebook
Data Format	CSV file

Extract, Transform, Load - ETL

- Load the data from the **UC Irvine Machine Learning Repository**

http://archive.ics.uci.edu/static/public/519/heart+failure+clinical+records.zip

```
In [12]: dat = pd.read_csv("heart_failure_clinical_records_dataset.csv", encoding='ISO-8859-1')
         dat.head()
```

Out[12]:

	age	anaemia	creatinine_phosphokinase	diabetes	ejection_fraction	high_blood_pressure	platelets	serum_creatinine	serum_sodium	sex	smoking	time	DEATH_EVENT
0	75.0	0	582	0	20	1	265000.00	1.9	130	1	0	4	1
1	55.0	0	7861	0	38	0	263358.03	1.1	136	1	0	6	1
2	65.0	0	146	0	20	0	162000.00	1.3	129	1	1	7	1
3	50.0	1	111	0	20	0	210000.00	1.9	137	1	0	7	1
4	65.0	1	160	1	20	0	327000.00	2.7	116	0	0	8	1

```
In [22]: import pandas as pd
         from pyspark.sql import SparkSession

         spark = SparkSession.builder.appName('heart+failure+clinical+records.zip').getOrCreate()
         datt = spark.read.csv('heart_failure_clinical_records_dataset.csv', header = True, inferSchema = True)
         datt.printSchema()
```

age	anaemia	creatinine_phosphokinase	diabetes	ejection_fraction	high_blood_pressure	platelets	serum_creatinine	serum_sodium	sex	smoking	time	DEATH_EVENT
75.0	0	582	0	20	1	265000.0	1.9	130	1	0	4	1
55.0	0	7861	0	38	0	263358.03	1.1	136	1	0	6	1
65.0	0	146	0	20	0	162000.0	1.3	129	1	1	7	1
50.0	1	111	0	20	0	210000.0	1.9	137	1	0	7	1
65.0	1	160	1	20	0	327000.0	2.7	116	0	0	8	1
90.0	1	47	0	40	1	204000.0	2.1	132	1	1	8	1
75.0	1	246	0	15	0	127000.0	1.2	137	1	0	10	1
60.0	1	315	1	60	0	454000.0	1.1	131	1	1	10	1
65.0	0	157	0	65	0	263358.03	1.5	138	0	0	10	1
80.0	1	123	0	35	1	388000.0	9.4	133	1	1	10	1

only showing top 10 rows

Exploratory Data Analysis (EDA)

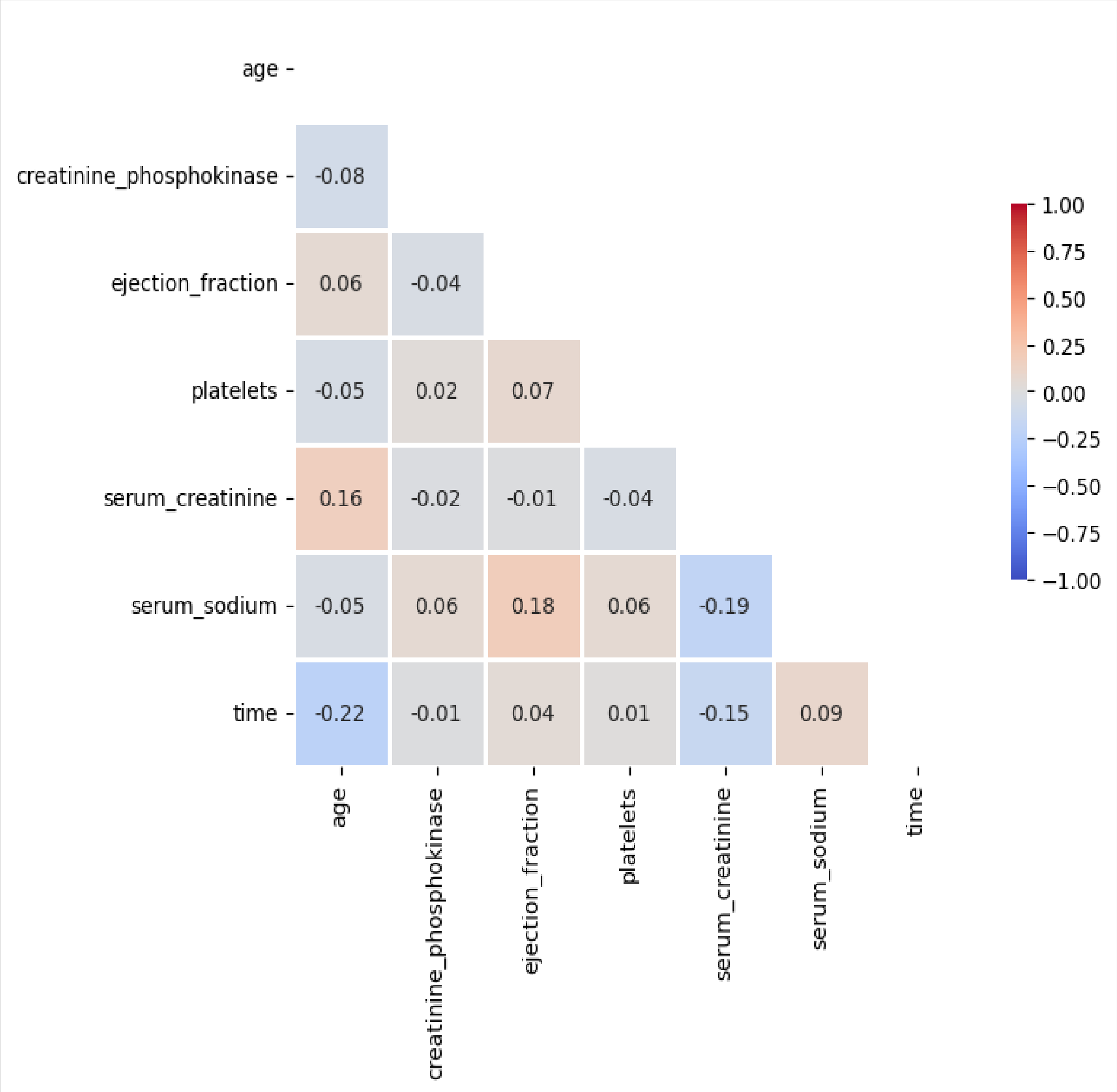
Assessment of Data Quality

Data	Explanation
Dimension	299 observations * 13 features
Data type	(int64, float64)
Missing value	No

```
age                         float64
anaemia                       int64
creatinine_phosphokinase      int64
diabetes                      int64
ejection_fraction             int64
high_blood_pressure           int64
platelets                   float64
serum_creatinine            float64
serum_sodium                  int64
sex                           int64
smoking                       int64
time                          int64
DEATH_EVENT                   int64
dtype: object
```

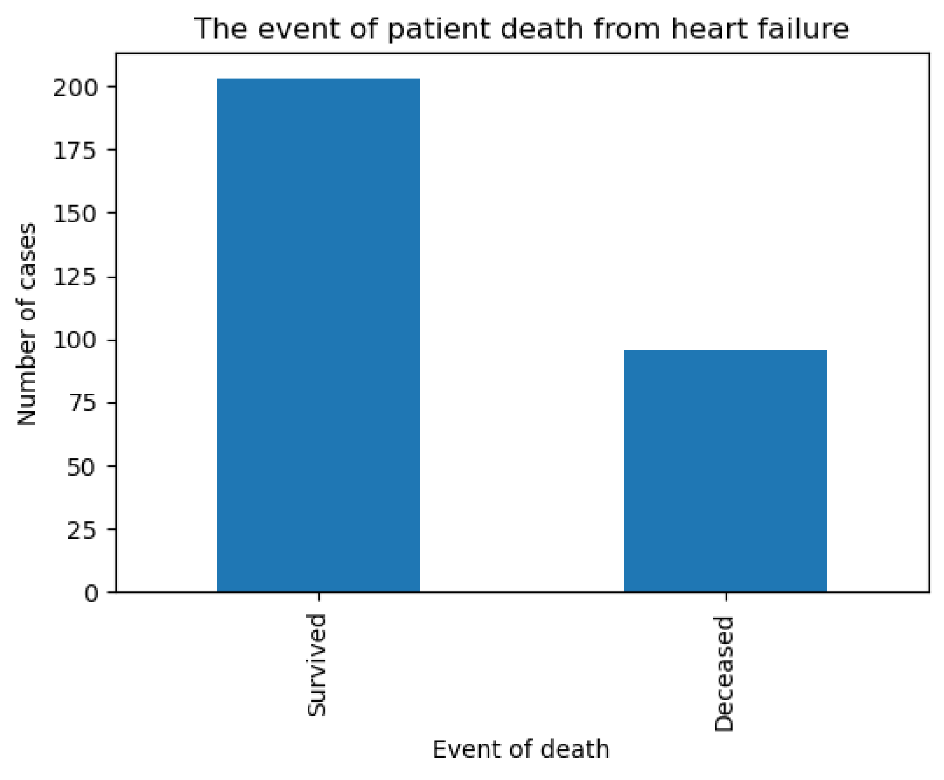
Data Exploration

- The correlation between numerical features is weak (≤ 0.22)
- The data is fit to run further modeling

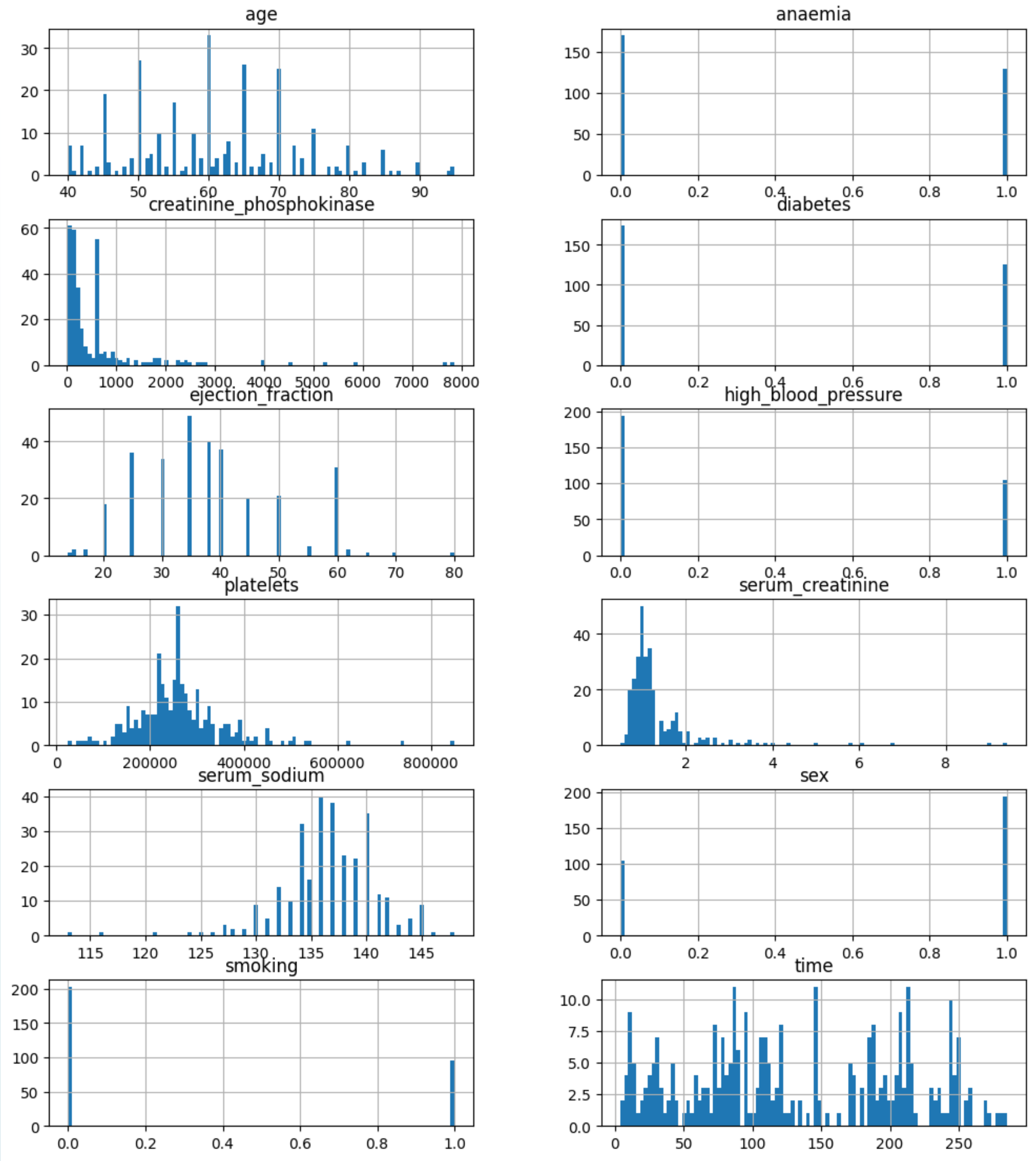


Data Visualization

DEATH_EVENT	count
1	96
0	203



Distribution of dependent variable (target)



Distribution of independent variable (input features)

Feature Engineering

- Five categorical integer features transform one-hot vectors using the OneHotEncoder function
- All input features transform into a single vector feature using the VectorAssembler function
- Then those input features normalize using the MinMaxScaler function
- Finally, two columns keep remained and it is ready for Machine Learning Models

```
+-----+--------------------+
|label|            features|
+-----+--------------------+
|    1|[1.0,1.0,0.0,0.0,...|
|    1|[1.0,1.0,1.0,0.0,...|
|    1|[1.0,1.0,1.0,0.0,...|
|    1|[0.0,1.0,1.0,0.0,...|
|    1|[0.0,0.0,1.0,1.0,...|
|    1|[0.0,1.0,0.0,0.0,...|
|    1|[0.0,1.0,1.0,0.0,...|
|    1|[0.0,0.0,1.0,0.0,...|
|    1|[1.0,1.0,1.0,1.0,...|
|    1|[0.0,1.0,0.0,0.0,...|
|    1|[0.0,1.0,0.0,0.0,...|
|    1|[1.0,1.0,0.0,0.0,...|
|    1|[0.0,1.0,1.0,0.0,...|
|    1|[0.0,1.0,0.0,0.0,...|
|    0|[0.0,1.0,0.0,1.0,...|
|    1|[0.0,1.0,1.0,0.0,...|
|    1|[0.0,1.0,1.0,0.0,...|
|    1|[1.0,1.0,1.0,0.0,...|
|    1|[0.0,1.0,0.0,1.0,...|
|    1|[0.0,0.0,1.0,1.0,...|
+-----+--------------------+
only showing top 20 rows
```

Model algorithms (Model definition and training)

- Labeled feature data Supervised Machine Learning
- The target - a binary categorical variable
- Classifier model

Machine Learning

- Supervised Machine Learning classifiers
    1. Logistic Regression
    2. Random Forest (RF) – bagging
    3. Gradient-Boosted Tree (GBT) – boosting

Deep Learning

- Feed Forward Neural Network/ Artificial Neural Network (ANN)
- Multi-layer Perceptron

Good for Classification and regression

Model performance indicators

- **Accuracy** score was used as a model performance indicator

Model accuracy for Logistic Regression – **79%**

```
In [33]: from pyspark.ml.classification import LogisticRegression
         lr = LogisticRegression().fit(dt_train)
         lr_smr = lr.summary
```

5.1.1 Model Evaluation for Logistic Regression

```
In [34]: ## Training data accuracy
         lr_smr.accuracy

Out[34]: 0.8565400843881856
```

```
In [37]: ## Model accuracy (Test data accuracy)
         model_predictions = lr.transform(dt_test)
         model_predictions = lr.evaluate(dt_test)

         model_predictions.accuracy

Out[37]: 0.7903225806451613
```

Model accuracy for Random Forest– **98%**

It seems overfitting in RF

5.2.1 Model Evaluation for Random Forest ML

In [46]:
```
from pyspark.ml.evaluation import MulticlassClassificationEvaluator
evaluator_rf = MulticlassClassificationEvaluator().setMetricName('accuracy').setPredictionCol('prediction').setLabelCol('label')
print("RF Training Accuracy")
print("--------------------")
evaluator_rf.evaluate(prediction_rf)
```

```
RF Training Accuracy
--------------------
```

Out[46]: 0.9957805907172996

In [47]:
```
print("RF Validation Accuracy")
print("--------------------")
accuracy = evaluator_rf.evaluate(prediction_rf_test)
accuracy
```

```
RF Validation Accuracy
--------------------
```

Out[47]: 0.9838709677419355

Model accuracy for Gradient-Boosted Tree – **100%**

5.3.1 Model Evaluation for Gradient-Boosted Trees ML

```
In [52]: from pyspark.ml.evaluation import MulticlassClassificationEvaluator

         evaluator = MulticlassClassificationEvaluator().setMetricName('accuracy').setPredictionCol('prediction').setLabelCol('label')
         print("GBT Training Accuracy")
         print("--------------------")
         evaluator.evaluate(prediction_gbt)

         GBT Training Accuracy
         --------------------
Out[52]: 0.9915611814345991

In [53]: print("GBT Validation Accuracy")
         print("--------------------")
         accuracy = evaluator.evaluate(prediction_gbt_test)
         accuracy

         GBT Validation Accuracy
         --------------------
Out[53]: 1.0
```

It has overfitting in GBT

Let's check model performance using Hyperparameter optimization (K-fold Cross Validation) for RF and GBT

Model accuracy for Feed Forward Neural Network – 79%

```
In [84]: # Evaluate the train
         loss, accuuracy =model_dl.evaluate(x_train,y_train)
         print('ANN Training Accuracy: {}\n Loss: {}'.format(accuuracy,loss))

8/8 [==============================] - 0s 1ms/step - loss: 0.3432 - accuracy: 0.8692
ANN Training Accuracy: 0.8691983222961426
 Loss: 0.3432324528694153

In [85]: ## Evaluate the model (test)
         loss, accuuracy =model_dl.evaluate(x_test,y_test)
         print('ANN Test Accuracy: {}\n Loss: {}'.format(accuuracy,loss))

2/2 [==============================] - 0s 2ms/step - loss: 0.4634 - accuracy: 0.7903
ANN Test Accuracy: 0.7903226017951965
 Loss: 0.46339651942253113
```

Model tuning

Random Forest accuracy after 5-fold CV – 85%

```
In [63]: # Use test set here so we can measure the accuracy of our best model
         rfpredictions1 = best_rf.transform(dt_test)

         ## Let's see the accuracy for test data
         print("RF Validation Accuracy (5-fold CrossValidation)")
         print("---------------------")
         eval.evaluate(rfpredictions1)

         RF Validation Accuracy (5-fold CrossValidation)
         ---------------------
Out[63]: 0.8478535353535352
```

Gradient-Boosted Tree accuracy after 5-fold CV – 84%

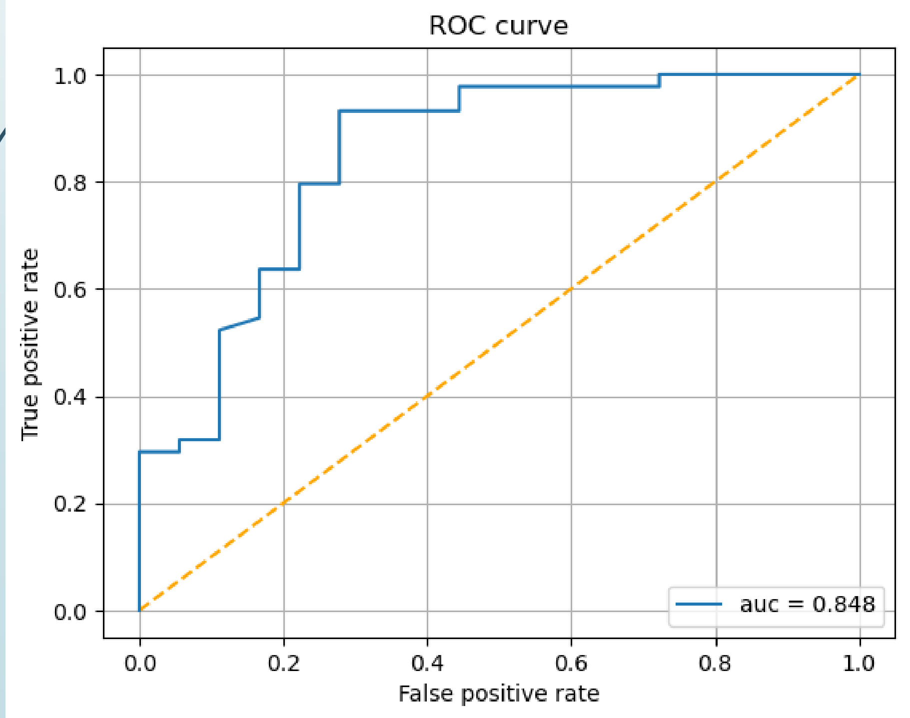
```
In [92]: # Use test set here so we can measure the accuracy of our model on new data
         gbtpredictions1 = best_gbt.transform(dt_test)

         ## Let's see the accuracy for test
         print("GBT Validation Accuracy for CrossValidation")
         print("--------------------")
         eval_gbt.evaluate(gbtpredictions1)

         GBT Validation Accuracy for CrossValidation
         --------------------
Out[92]: 0.8402777777777778
```

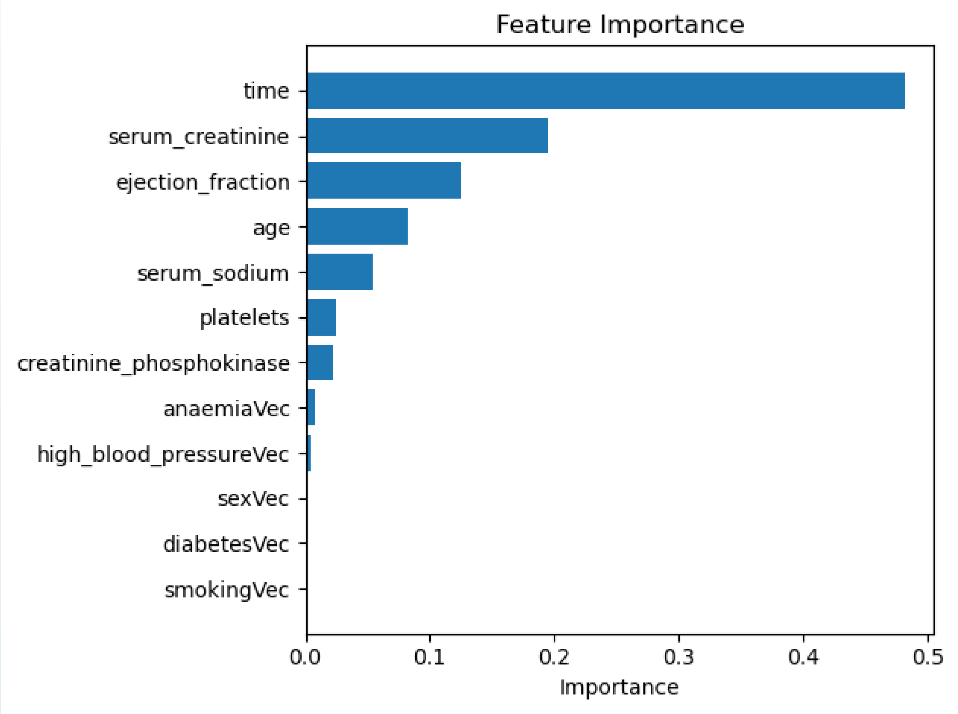
Random Forest Algorithm

After modeling three Machine Learning and one Deep Learning

Random Forest model performs better than other algorithms

	precision	recall	f1-score	support
0	0.84	0.93	0.88	44
1	0.77	0.56	0.65	18
accuracy			0.82	62
macro avg	0.80	0.74	0.76	62
weighted avg	0.82	0.82	0.81	62





References

Project GIST link (Jupyter Notebook)

IBM Watson Studio Notebook permalinks GIST
https://dataplatform.cloud.ibm.com/analytics/notebooks/v2/a4eedece-d914-4691-872b-8cc8c3f1cc08/view?access_token=09fa7105a248a3d7baf512bf4ad3a24578caf45a610e8116ae664a1cebb7dc71&context=cpdaas

Github link for the project

https://github.com/Zawmin2004/Advanced-Data-Science-with-IBM---Capstone-Project/blob/main/Capstone_Project_for_IBM_Advanced_Data_Science.ipynb

Architectural Decisions Document (ADD)

https://github.com/Zawmin2004/Advanced-Data-Science-with-IBM---Capstone-Project/blob/main/Architectural%20Decisions%20Design%20(ADD).pdf

Original paper

Chicco, D., Jurman, G. Machine learning can predict survival of patients with heart failure from serum creatinine and ejection fraction alone. BMC Med Inform Decis Mak 20, 16 (2020). https://doi.org/10.1186/s12911-020-1023-5

Thank you for your kind attention!