HEART DISEASE IDENTIFICATION FROM PATIENTS' SOCIAL POSTS, MACHINE LEARNING SOLUTION ON SPARK

Report submitted to the SASTRA Deemed to be University as the requirement for the course

BICCIC707: **MINI PROJECT**

Submitted by

MEKALA SAI YESHWANTH
(Reg. No.: 121014061, ICT)
CHITTIMADUGULA SAMBA SIVA RAO
(Reg. No.: 121014013, ICT)
KADIRIMANGALAM SADHWIK
(Reg. No.: 120014018, ICT)

December 2020

SCHOOL OF COMPUTING

THANJAVUR, TAMIL NADU, INDIA – 613 401

SCHOOL OF COMPUTING

THANJAVUR – 613 401

Bonafide Certificate

This is to certify that the report titled "Heart Disease Identification from Patients' Social Posts, Machine Learning Solution on Spark" submitted as a requirement for the course, BICCIC707**:** **MINI PROJECT** for B.Tech. is a bonafide record of the work done by **Mr. KADIRIMANGALAM SADHWIK (Reg.No.120014018,ICT**), **Mr.CHITTIMADUGULA SAMBA SIVA RAO (Reg. No.121014013, ICT**),**Mr. MEKALA SAI YESHWANTH (Reg. No. 121014061, ICT**) during the academic year 2020-21, in the School of Computing, under my supervision.

Signature of Project Supervisor:
Name with Affiliation : Dr. V. Subramaniyaswamy, Associate Professor, SOC, SASTRA
Date : 23-13-2020

Mini Project *Viva voce* held on _____

Examiner 1 **Examiner 2**

ACKNOWLEDGEMENT

We would forever remain grateful to honorable Dr. S. Vaidhyasubramaniam, Vice Chancellor for his encouragement in our academic life at SASTRA Deemed to be University. We wish to express our profound gratitude to Dr. R. Chandramouli, Registrar for their overwhelming support provided during our course span in SASTRA University. We are extremely thankful to Dr. A. Umamakeswari, Dean School of Computing and Dr. V. S. Shankar Sriram, Associate Dean, Department of Computer Science Technology for providing us the opportunity to do this project and for all the academic help extended in our project.

We sincerely express our gratitude to our Guide Dr. V. Subramaniyaswamy, Associate Professor, SOC, SASTRA for his assistance and guidance for the successful implementation of the project in a systematic and professional manner.

Table of Contents

Bonafide Certificate **ii**

ACKNOWLEDGEMENT **iii**

LIST OF FIGURES **v**

LIST OF TABLES **vi**

ABBREVIATIONS **vii**

ABSTRACT **viii**

1. SUMMARY OF THE BASE PAPER **1**

1.1 Introduction 1

1.2 Methodology 2

1.2.1 Building Offline Model 2

1.2.2 Streaming Process Pipeline 4

1.2.3 Online Prediction 4

1.3 Dataset 4

1.3.1 The Cleveland Heart Disease Data set 5

1.3.2 Real-time twitter data 6

2. MERITS AND DEMERITS **7**

2.1 Literature Review 7

2.2 Merits and Demerits 7

3. SOURCE CODE **8**

4. SNAPSHOTS **19**

4.1 Univariate Feature Selection 19

4.2 Relief Feature Selection 20

4.3 Accuracy Comparison 21

4.4 Best Parameters Values 22

4.5 Online Prediction Results 23

5. CONCLUSIONS AND FUTURE PLANS **25**

6. REFERENCES **26**

7. PLAGIARISM REPORT **27**

8. APPENDIX-BASE PAPER **28**

LIST OF FIGURES

Figure No.	Title	Page No.
Fig. 1.1	Architecture of the Prediction System	2
Fig. 1.2	The Architecture of Building Offline Model	2
Fig 1.3	Description of the Features	5
Fig. 4.1	Univariate Feature Selection Scores	19
Fig. 4.2	Relief Feature Selection Ranking	20
Fig. 4.3	Accuracy Comparison of Full Features	21
Fig. 4.4	Accuracy Comparison of Univariate Features	21
Fig. 4.5	Accuracy Comparison of Relief Features	22
Fig. 4.6	Online Prediction Results	24

LIST OF TABLES

Table No.	Table Name	Page No.
Table 4.1	Selected Univariate Features with score	19
Table 4.2	Selected Relief Features with ranking	20
Table 4.3	Best Parameters values- Full Features	22
Table 4.4	Best Parameters values- Univariate Features	23
Table 4.5	Best Parameters values- Relief Features	23

ABBREVIATIONS

AHP	Analytic Hierarchy Process
ANFIS	Adaptive Neuro Fuzzy Inference System
API	Application Programming Interface
DT	Decision Tree
GA	Genetic Algorithm
JSON	JavaScript Object Notation
KEGG	Kyoto Encyclopedia of Genes and Genomes
LR	Logistic Regression
MATLAB	Matrix Laboratory
MKL	Multiple Kernel Learning
ML	Machine Learning
MSE	Mean Squared Error
RF	Random Forest
SCIE	Science Citation Index Expanded
SVM	Support Vector Machine

ABSTRACT

Heart disease is ruling the charts of deadly diseases all over the world. The work done in this paper helps to predict the heart disease in real-time from medical data, which depict the health status of an individual. This system proposed on Apache Spark finds the machine learning algorithm with best possible accuracy for prediction of heart disease on heart dataset from the UCI repository. Important features are selected from the dataset using univariate and relief feature selection techniques. we compared the accuracy of four machine learning classifiers specifically Logistic Regression, Decision Tree, Random Forest as well as Support Vector Machine Classifier on full features and selected features. Accuracy is improved by applying hyperparameter tuning and cross-validation. Random Forest Classifier has achieved the highest accuracy of 89.29% on full features. The most significant aspect of the proposed system is to prognosticate heart disease efficiently on the twitter data containing patient's data in real-time. Aforementioned is achieved by affiliating Apache Spark with Apache Kafka.

KEYWORDS: Machine Learning, Apache Spark, Apache Kafka

1. SUMMARY OF THE BASE PAPER

Title: Heart Disease Identification from Patients' Social Posts, Machine Learning Solution on Spark.

Journal Name: Future Generation Computer System

Publisher: ELSEVIER

Year: 2019

Indexed In: SCIE

1.1 Introduction

Heart disease is ruling the charts of the most eminent causes of death all over the world. Heart diseases allude to various contexts that affect proper functioning of heart. In the USA, around 17.9 million people deceased due to heart disease in 2016 [1]. Heart diseases like stroke, hypertension are very dangerous and lead to death.

So, a system with high accuracy is needed for early prediction of heart disease by understanding the patterns that causes heart diseases. This helps in early detection of the attack and the patient can be treated accordingly. Many systems based on various machine learning and deep learning classification algorithm with fairly good accuracy have been proposed. Due to rapid growth of population, the medical records are also increasing exponential. It is tough to handle those huge amount of medical data. Many researches are going on regarding this problem. One of the most challenging tasks is to prognosticate heart disease in real-time. The most significant aspect of this paper is real-time prediction of tweets. The possibility of heart disease in streaming data is prognosticated in real-time by integrating Apache Spark and Apache Kafka.

The motive of the work is deploying a model with best possible accuracy on Cleveland heart disease dataset 2016 from UCI repository [2] and then by using this best model the streaming medical data from twitter is predicted. This System incorporates three components: 1. Building Offline Model, 2. Streaming Process Pipeline, and 3. Online prediction. Aforementioned is built on Apache Spark and Apache Kafka which helps in prediction of huge amount of data. The offline model is built by using four machine learning algorithms LR, DT, RF, and SVM on full features and selected features by using Univariate and Relief feature selection algorithms. Accuracy is improved by applying hyperparameter tuning and k-fold cross-validation techniques. RF has achieved the highest accuracy of 89.9%.

1.2 Methodology

The architecture comprises three components specifically 1. Building Offline Model, 2. Streaming Process Pipeline, and 3. Online Prediction, depicted in Fig. 1.1 below

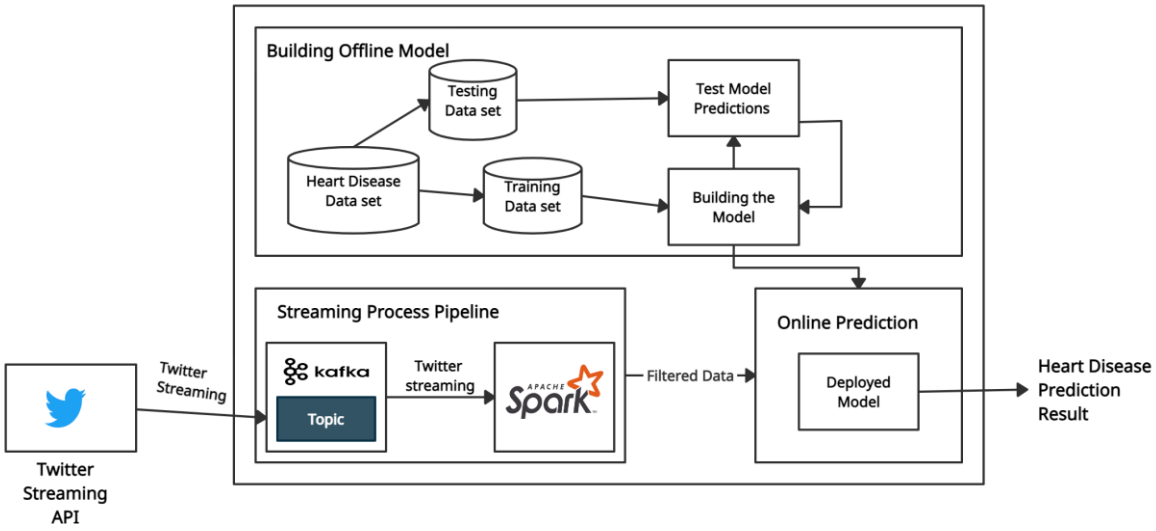

Fig. 1.1 Architecture of the Prediction System

1.2.1 Building Offline Model

The present component aims to build ML models and find the model with high accuracy. This component comprises four stages namely A. Data preprocessing, B. Feature Selection, C. Applying ML Classifiers, and D. Evaluating the model as depicted in Fig. 1.2 below

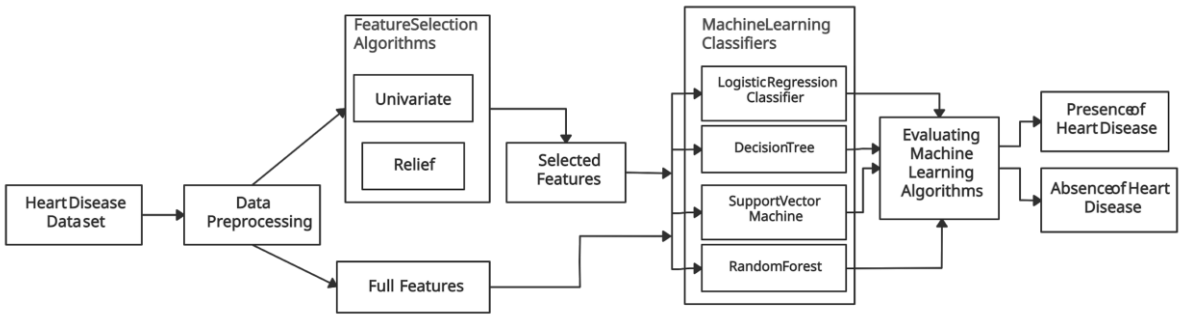

Fig. 1.2 The Architecture of Building Offline Model

A. Data processing

Missing values in the dataset are removed if exists, then MinMaxScaler is applied to scale all the values of features to the range of 0 and 1.

B. Feature Selection

Important features from the preprocessed dataset are selected by using I. Univariate and II. Relief feature selection techniques.

I. Univariate Feature Selection

The top seven features with the strongest relation with class label are selected using this statistical test. SelectKBest with chi-squared statistic test has been utilized to select the features with highest values.

II. Relief Feature Selection

Initially zero weight is assigned to all the features, then the weights keep updating with time. The top seven features with highest weights were selected.

C. Applying ML Classifiers

Classifiers mentioned underneath have been used in this stage:

I. Logistic Regression

LR is a binary classifier that predicts values either 0 or 1, where 0,1 represents positive and negative classes respectively.

II. Decision Tree

DT is a collection of nodes. In this, features are represented as internal nodes and class label as leaf nodes and the result is represented as a branch. It is iterative and the dataset is splitted based on either of the two criterions namely information gain and entropy.

III. Random Forest

RF is an ensemble algorithm. It consists of a collection of decision trees. A unit vote will be given by each tree which represents the most popular class variable that will classify the input vector.

IV. SVM

It is a supervised algorithm, which performs classification as well as regression. It identifies the best hyperplane, which classifies the data into class 0 or 1 by separating the variables.

D. Evaluating the model

In this stage, the accuracy of the developed models is increased by doing the following steps:

I. Hyperparameter tuning

In this step, different parameters values are passed using grid search. Each hyperparameter values will be tested by the model and finally we will select the best parameters which will give the highest accuracy.

II. K-Fold Cross-Validation

This will split the dataset into k similar size folds. Each fold will be utilized to test the model and final accuracy will be calculated by taking the average of the accuracy acquired by k folds.10 folds have been used in this paper, where 90% of the dataset will be utilized for training and remaining 10% for testing the model.

1.2.2 Streaming Process Pipeline

Apache Spark is affiliated with Apache Kafka for evaluating the model in real-time.

I. Apache Kafka

It is used to receive huge amounts of data with very low latency in real-time and store them in Kafka Topic. It retrieves Tweets with header "hrtdis" and stores in Kafka Topic. It creates the pipeline with Apache Spark. It consists of Producer and Consumer API. Here, Twitter Streaming API will act as producer where the tweets will be generated and Apache Spark will act as Consumer by accepting the data streams from Kafka Topic.

II. Apache Spark

It is an open-source Big Data framework which helps in processing the data. It reads data streams from Kafka Topic in JSON format with help of Spark Streaming API. By using Spark libraries data will be preprocessed by removing unwanted data and converts it into vector.

1.2.3 Online Prediction

The selected model with highest accuracy (RF) will receive the data in the form of a vector from Apache Spark and will prognosticate the heart disease.

1.3 Dataset

This section broadly describes the outline of the dataset that was used in Building Offline Model and also about the details of the dataset that was used for real-time evaluation of the system.

1.3.1 The Cleveland Heart Disease Data set

The outline of the dataset used to build the offline model component is discussed underneath. The data set i.e., processed Cleveland data is an open-source data set [2], which is available in UCI machine learning repository. The data set consists of 303 rows and 14 columns. The data set mentioned above contains 13 independent features and a dependent class label variable. The class label column in the data set depicts the status of the heart disease. The description of features of the dataset is presented in Fig. 1.3 below. The class label variable holds five values, where a value 0 represents the deprivation of heart disease whereas the residual values represent a possible existence of heart disease. The work carried out on the dataset is binary class classification. So, the values in the class label are scaled to hold either 0 or 1, where he value 0 holds its original description and value 1 is a generalization of the residual values which indicates the existence of heart disease.

S. No.	Feature Name	Code
1	Age	AGE
2	Sex	SEX
3	Type of chest pain	CPT
4	Resting blood pressure	RBP
5	Serum cholesterol	SCH
6	Fasting blood sugar	FBS
7	Resting electrocardiographic results	RES
8	Maximum heart rate achieved	MHR
9	Exercise-induced angina	EIA
10	Old peak	OPK
11	Slope of the peak exercise ST segment	PES
12	Number of major vessels (0-3) colored by fluoroscopy	VCA
13	Thallium scan	THA

Fig 1.3 Description of the Features

1.3.2 Real-time twitter data

Twitter is a most popular social network platform and is widely used to share, post content and to discuss various topics. Hence, twitter is recognized a best pedigree for real-time health data. Users tweet the heart related tweets which constitute to the stream of tweets that are to be retrieved from twitter. The filtered data from the tweets is used for the real-time evaluation of the proposed prediction system. For the proposed system, to retrieve tweet streams from twitter, it should establish an authorized connection with Twitter. This connection is made using the Twitter streaming API. Twitter uses a protocol called OAuth for an authenticated connection and security purposes, to access the services from twitter and use the data. To make this authenticated connection we should apply for a developer account on twitter. On a successful approval for a developer account, the user will be issued unique keys, which can be used to establish an authenticated connection and access the tweet streams. After the connection is established, tweets with the header “hrtdis” are retrieved. The tweet consists of a sequence of values which appear in the same sequence as in the training data.

2. MERITS AND DEMERITS

2.1 Literature Review

Gunasekaran Manogaran et al. [3] introduced MKL combined by ANFIS. A KEGG Metabolic Reaction Network dataset was used to test the system and produced high specificity of 99%, high sensitivity of 98% and MSE was 0.01.

Nazari et al. [4] developed a technique with Fuzzy AHP as well as Fuzzy Inference System. Weights for several criterion that impact the developing heart disease is calculated using Fuzzy AHP. Evaluation of chances of developing heart disease is calculated using Fuzzy Inference System. The system was trained and tested with a dataset in a hospital in Tehran.

Jayaraman et al. [5] introduced Artificial Gravitational Cuckoo Search Algorithm combined with Particle Bee Optimized Associative Memory Neural Network approach for selecting important features. Heart dataset from UCI repository was used and the efficiency was calculated using MATLAB based experimental results.

Gokulnath et al. [6] proposed Support Vector Machine (SVM) based optimization function used in Genetic Algorithm (GA) in MATLAB environment. Most important features are selected using GA. Cleveland heart disease dataset was used and the accuracy of SVM classifier was 83.70% for 13 features. After using GA the accuracy was increased to 88.34%.

2.2 Merits and Demerits

In this paper, a system depend on Apache Spark affiliated with Apache Kafka was developed to predict in real-time if the patient is prognosticated in heart disease. Hyperparameter tuning as well as K-Fold Cross-Validation were used to enhance accuracy and subsequent outcomes reflect that the accuracy of RF (89.9%) was better in contrast to the competitive work.

3. SOURCE CODE

```
#removing missing values
heartds.replace('?', np.nan, inplace=True)
heartds.dropna()

#MinMax Scaler
from sklearn.preprocessing import MinMaxScaler
scaler=MinMaxScaler()
scaled_heartds=pd.DataFrame(scaler.fit_transform(heartds),index=heartds.index,columns
=heartds.columns)
print(scaled_heartds.head())
scaled_heartds.dtypes

#vector conversion
from pyspark.ml.feature import VectorAssembler
cols=heartds.columns
cols.remove('target')
assembler=VectorAssembler(inputCols=cols,outputCol='features')
heartds=assembler.transform(heartds)
heartds.select('features').show(truncate=False)

#Univariate Feature Selection
unifeatures = SelectKBest(chi2,k=7).fit(x_train,y_train)
print('chi2 scores are:')
for i in range(len(unifeatures.scores_)):
  print('%s: %f'%(x.columns[i],unifeatures.scores_[i]))
print('7 important features are:')
uf=[]
for i,j in zip(unifeatures.get_support(),x.columns):
  if i:
    uf.append(j)
print(uf)

#Relief Feature Selection
!pip install skrebate
from skrebate import ReliefF
x_train=x_train.astype('float')
relfeatures=ReliefF(n_features_to_select=7,n_neighbors=100).fit(x_train,y_train)
header=train.columns.to_list()
features=header[0:len(header)-1]
weights={'name':features,'weights':relfeatures.feature_importances_}
weights_df=pd.DataFrame(weights)
```

```
weights_df=weights_df.sort_values(by='weights')
print('weights are:')
print(weights_df)
rf=list(weights_df['name'].loc[7:])[::-1]
print('7 important features are:')
print(rf)

#Splitting dataset
train,test=heartds.randomSplit([0.9,0.1])

#Accuracy function
def accuracy(pred):
  pred.groupBy('target','prediction').count().show()
  tn=pred.filter('prediction=0 AND target=prediction').count()
  tp=pred.filter('prediction=1 AND target=prediction').count()
  fn=pred.filter('prediction=0 AND target<>prediction').count()
  fp=pred.filter('prediction=1 AND target<>prediction').count()
  acc=((tn+tp)/(tn+tp+fn+fp))*100
  print('Accuracy=%0.2f'%acc)

#Logistic Regression on Full Features
from pyspark.ml.classification import LogisticRegression
lr=LogisticRegression(labelCol='target',featuresCol='scaledfeatures')
lrmodel=lr.fit(train)
lrpredict_train=lrmodel.transform(train)
lrpredict_test=lrmodel.transform(test)
accuracy(lrpredict_test)
print('regParam:',lrmodel.getOrDefault('regParam'))
print('maxIter:',lrmodel.getOrDefault('maxIter'))

evaluator=BinaryClassificationEvaluator(rawPredictionCol='rawPrediction',labelCol='tar
get')
paramgrid=ParamGridBuilder().addGrid(lr.regParam,[0.1,0.01,0.2,0.3,0.4,0.5]).addGrid(l
r.maxIter,[0,10,30,40,50,75,100]).build()
cv=CrossValidator(estimator=lr,estimatorParamMaps=paramgrid,evaluator=evaluator,nu
mFolds=10)
cvmodel=cv.fit(train)
lrpredict_train=cvmodel.transform(train)
lrpredict_test=cvmodel.transform(test)
bestlrmodel=cvmodel.bestModel
accuracy(lrpredict_test)
print('regParam:',bestlrmodel.getOrDefault('regParam'))
print('maxIter:',bestlrmodel.getOrDefault('maxIter'))
```

```
#Decision Tree on Full Features
from pyspark.ml.classification import DecisionTreeClassifier
dt=DecisionTreeClassifier(labelCol='target',featuresCol='scaledfeatures')
dtmodel=dt.fit(train)
dtpredict_train=dtmodel.transform(train)
dtpredict_test=dtmodel.transform(test)
accuracy(dtpredict_test)
print('maxdepth:',dtmodel.getOrDefault('maxDepth'))
print('maxBins:',dtmodel.getOrDefault('maxBins'))
print('impurity:',dtmodel.getOrDefault('impurity'))

evaluator=BinaryClassificationEvaluator(rawPredictionCol='rawPrediction',labelCol='tar
get')
paramgrid=ParamGridBuilder().addGrid(dt.maxDepth,[2,3,4,5]).addGrid(dt.impurity,['ent
ropy','gini']).addGrid(dt.maxBins,[10,15,20,30,35,40]).build()
cv=CrossValidator(estimator=dt,estimatorParamMaps=paramgrid,evaluator=evaluator,nu
mFolds=10)
cvmodel=cv.fit(train)
dtpredict_train=cvmodel.transform(train)
dtpredict_test=cvmodel.transform(test)
bestdtmodel=cvmodel.bestModel
accuracy(dtpredict_test)
print('maxdepth:',bestdtmodel.getOrDefault('maxDepth'))
print('maxBins:',bestdtmodel.getOrDefault('maxBins'))
print('impurity:',bestdtmodel.getOrDefault('impurity'))

#Random Forest on Full Features
from pyspark.ml.classification import RandomForestClassifier
rf=RandomForestClassifier(labelCol='target',featuresCol='scaledfeatures')
rfmodel=rf.fit(train)
rfpredict_train=rfmodel.transform(train)
rfpredict_test=rfmodel.transform(test)
accuracy(rfpredict_test)
print('maxdepth:',rfmodel.getOrDefault('maxDepth'))
print('maxBins:',rfmodel.getOrDefault('maxBins'))
print('numTrees:',rfmodel.getOrDefault('numTrees'))

from pyspark.ml.evaluation import BinaryClassificationEvaluator
from pyspark.ml.tuning import ParamGridBuilder,CrossValidator
evaluator=BinaryClassificationEvaluator(rawPredictionCol='rawPrediction',labelCol='tar
get')
```

```
paramgrid=ParamGridBuilder().addGrid(rf.maxDepth,[2,3,4,5]).addGrid(rf.maxBins,[10,
15,20,30,35,40]).addGrid(rf.numTrees,[20,30,40,45]).build()
cv=CrossValidator(estimator=rf,estimatorParamMaps=paramgrid,evaluator=evaluator,nu
mFolds=10)
cvmodel=cv.fit(train)
rfpredict_train=cvmodel.transform(train)
rfpredict_test=cvmodel.transform(test)
bestrfmodel=cvmodel.bestModel
accuracy(rfpredict_test)
print('Best maxdepth:',bestrfmodel.getOrDefault('maxDepth'))
print('Best maxBins:',bestrfmodel.getOrDefault('maxBins'))
print('best numTrees:',bestrfmodel.getOrDefault('numTrees'))

#SVM on Full Features
from pyspark.ml.classification import LinearSVC
sv=LinearSVC(labelCol='target',featuresCol='scaledfeatures')
svmodel=sv.fit(train)
svpredict_train=svmodel.transform(train)
svpredict_test=svmodel.transform(test)
accuracy(svpredict_test)
print('regParam:',svmodel.getOrDefault('regParam'))
print('maxIter:',svmodel.getOrDefault('maxIter'))

evaluator=BinaryClassificationEvaluator(rawPredictionCol='rawPrediction',labelCol='tar
get')
paramgrid=ParamGridBuilder().addGrid(sv.regParam,[0.01,0.02,0.03,0.1,0.2,0.3]).addGri
d(sv.maxIter,[10,20,25,30,35,40,45,50]).build()
cv=CrossValidator(estimator=sv,estimatorParamMaps=paramgrid,evaluator=evaluator,nu
mFolds=10)
cvmodel=cv.fit(train)
svpredict_train=cvmodel.transform(train)
svpredict_test=cvmodel.transform(test)
bestsvmodel=cvmodel.bestModel
accuracy(svpredict_test)
print('regParam:',bestsvmodel.getOrDefault('regParam'))
print('maxIter:',bestsvmodel.getOrDefault('maxIter'))

#LR on Univariate Features
lr=LogisticRegression(labelCol='target',featuresCol='scaledunifeatures')
lrmodel=lr.fit(train)
lrpredict_train=lrmodel.transform(train)
lrpredict_test=lrmodel.transform(test)
accuracy(lrpredict_test)
```

```
print('regParam:',lrmodel.getOrDefault('regParam'))
print('maxIter:',lrmodel.getOrDefault('maxIter'))

evaluator=BinaryClassificationEvaluator(rawPredictionCol='rawPrediction',labelCol='tar
get')
paramgrid=ParamGridBuilder().addGrid(lr.regParam,[0.1,0.01,0.2,0.3,0.4,0.5]).addGrid(l
r.maxIter,[0,10,30,40,50,75,100]).build()
cv=CrossValidator(estimator=lr,estimatorParamMaps=paramgrid,evaluator=evaluator,nu
mFolds=10)
cvmodel=cv.fit(train)
lrpredict_train=cvmodel.transform(train)
lrpredict_test=cvmodel.transform(test)
bestlrmodel=cvmodel.bestModel
accuracy(lrpredict_test)
print('regParam:',bestlrmodel.getOrDefault('regParam'))
print('maxIter:',bestlrmodel.getOrDefault('maxIter'))

#DT on Univariate Features
dt=DecisionTreeClassifier(labelCol='target',featuresCol='scaledunifeatures')
dtmodel=dt.fit(train)
dtpredict_train=dtmodel.transform(train)
dtpredict_test=dtmodel.transform(test)
accuracy(dtpredict_test)
print('maxdepth:',dtmodel.getOrDefault('maxDepth'))
print('maxBins:',dtmodel.getOrDefault('maxBins'))
print('impurity:',dtmodel.getOrDefault('impurity'))

evaluator=BinaryClassificationEvaluator(rawPredictionCol='rawPrediction',labelCol='tar
get')
paramgrid=ParamGridBuilder().addGrid(dt.maxDepth,[2,3,4,5]).addGrid(dt.impurity,['ent
ropy','gini']).addGrid(dt.maxBins,[10,15,20,30,35,40]).build()
cv=CrossValidator(estimator=dt,estimatorParamMaps=paramgrid,evaluator=evaluator,nu
mFolds=10)
cvmodel=cv.fit(train)
dtpredict_train=cvmodel.transform(train)
dtpredict_test=cvmodel.transform(test)
bestdtmodel=cvmodel.bestModel
accuracy(dtpredict_test)
print('maxdepth:',bestdtmodel.getOrDefault('maxDepth'))
print('maxBins:',bestdtmodel.getOrDefault('maxBins'))
print('impurity:',bestdtmodel.getOrDefault('impurity'))
```

```
#RF on Univariate Features
rf=RandomForestClassifier(labelCol='target',featuresCol='scaledunifeatures')
rfmodel=rf.fit(train)
rfpredict_train=rfmodel.transform(train)
rfpredict_test=rfmodel.transform(test)
accuracy(rfpredict_test)
print('maxdepth:',rfmodel.getOrDefault('maxDepth'))
print('maxBins:',rfmodel.getOrDefault('maxBins'))
print('numTrees:',rfmodel.getOrDefault('numTrees'))

evaluator=BinaryClassificationEvaluator(rawPredictionCol='rawPrediction',labelCol='tar
get')
paramgrid=ParamGridBuilder().addGrid(rf.maxDepth,[2,3,4,5]).addGrid(rf.maxBins,[10,
15,20,30,35,40]).addGrid(rf.numTrees,[20,30,35,40]).addGrid(rf.impurity,['entropy','gini'
]).build()
cv=CrossValidator(estimator=rf,estimatorParamMaps=paramgrid,evaluator=evaluator,nu
mFolds=10)
cvmodel=cv.fit(train)
rfpredict_train=cvmodel.transform(train)
rfpredict_test=cvmodel.transform(test)
bestrfmodel=cvmodel.bestModel
accuracy(rfpredict_test)
print('Best maxdepth:',bestrfmodel.getOrDefault('maxDepth'))
print('Best maxBins:',bestrfmodel.getOrDefault('maxBins'))
print('best numTrees:',bestrfmodel.getOrDefault('numTrees'))

#SVM on Univariate Features
sv=LinearSVC(labelCol='target',featuresCol='scaledunifeatures')
svmodel=sv.fit(train)
svpredict_train=svmodel.transform(train)
svpredict_test=svmodel.transform(test)
accuracy(svpredict_test)
print('regParam:',svmodel.getOrDefault('regParam'))
print('maxIter:',svmodel.getOrDefault('maxIter'))

evaluator=BinaryClassificationEvaluator(rawPredictionCol='rawPrediction',labelCol='tar
get')
paramgrid=ParamGridBuilder().addGrid(sv.regParam,[0.01,0.02,0.03,0.1,0.2,0.3]).addGri
d(sv.maxIter,[10,20,25,30,35,40,45,50]).build()
cv=CrossValidator(estimator=sv,estimatorParamMaps=paramgrid,evaluator=evaluator,nu
mFolds=10)
cvmodel=cv.fit(train)
svpredict_train=cvmodel.transform(train)
```

```
svpredict_test=cvmodel.transform(test)
bestsvmodel=cvmodel.bestModel
accuracy(svpredict_test)
print('regParam:',bestsvmodel.getOrDefault('regParam'))
print('maxIter:',bestsvmodel.getOrDefault('maxIter'))

#LR on Relief Features
lr=LogisticRegression(labelCol='target',featuresCol='scaledrelieffeatures')
lrmodel=lr.fit(train)
lrpredict_train=lrmodel.transform(train)
lrpredict_test=lrmodel.transform(test)
accuracy(lrpredict_test)
print('regParam:',lrmodel.getOrDefault('regParam'))
print('maxIter:',lrmodel.getOrDefault('maxIter'))

evaluator=BinaryClassificationEvaluator(rawPredictionCol='rawPrediction',labelCol='tar
get')
paramgrid=ParamGridBuilder().addGrid(lr.regParam,[0.1,0.01,0.2,0.3,0.4,0.5]).addGrid(l
r.maxIter,[0,10,30,40,50,75,100]).build()
cv=CrossValidator(estimator=lr,estimatorParamMaps=paramgrid,evaluator=evaluator,nu
mFolds=10)
cvmodel=cv.fit(train)
lrpredict_train=cvmodel.transform(train)
lrpredict_test=cvmodel.transform(test)
bestlrmodel=cvmodel.bestModel
accuracy(lrpredict_test)
print('regParam:',bestlrmodel.getOrDefault('regParam'))
print('maxIter:',bestlrmodel.getOrDefault('maxIter'))

#DT on Relief Features
dt=DecisionTreeClassifier(labelCol='target',featuresCol='scaledrelieffeatures')
dtmodel=dt.fit(train)
dtpredict_train=dtmodel.transform(train)
dtpredict_test=dtmodel.transform(test)
accuracy(dtpredict_test)
print('maxdepth:',dtmodel.getOrDefault('maxDepth'))
print('maxBins:',dtmodel.getOrDefault('maxBins'))
print('impurity:',dtmodel.getOrDefault('impurity'))

evaluator=BinaryClassificationEvaluator(rawPredictionCol='rawPrediction',labelCol='tar
get')
paramgrid=ParamGridBuilder().addGrid(dt.maxDepth,[2,3,4,5]).addGrid(dt.impurity,['ent
ropy','gini']).addGrid(dt.maxBins,[10,15,20,30,35,40]).build()
```

```
cv=CrossValidator(estimator=dt,estimatorParamMaps=paramgrid,evaluator=evaluator,numFolds=10)
cvmodel=cv.fit(train)
dtpredict_train=cvmodel.transform(train)
dtpredict_test=cvmodel.transform(test)
bestdtmodel=cvmodel.bestModel
accuracy(dtpredict_test)
print('maxdepth:',bestdtmodel.getOrDefault('maxDepth'))
print('maxBins:',bestdtmodel.getOrDefault('maxBins'))
print('impurity:',bestdtmodel.getOrDefault('impurity'))

#RF on Relief Features
rf=RandomForestClassifier(labelCol='target',featuresCol='scaledrelieffeatures')
rfmodel=rf.fit(train)
rfpredict_train=rfmodel.transform(train)
rfpredict_test=rfmodel.transform(test)
accuracy(rfpredict_test)
print('maxdepth:',rfmodel.getOrDefault('maxDepth'))
print('maxBins:',rfmodel.getOrDefault('maxBins'))
print('numTrees:',rfmodel.getOrDefault('numTrees'))

evaluator=BinaryClassificationEvaluator(rawPredictionCol='rawPrediction',labelCol='target')
paramgrid=ParamGridBuilder().addGrid(rf.maxDepth,[2,3,4,5]).addGrid(rf.maxBins,[10,15,20,30,35,40]).addGrid(rf.numTrees,[20,30,35,40]).addGrid(rf.impurity,['entropy','gini']).build()
cv=CrossValidator(estimator=rf,estimatorParamMaps=paramgrid,evaluator=evaluator,numFolds=10)
cvmodel=cv.fit(train)
rfpredict_train=cvmodel.transform(train)
rfpredict_test=cvmodel.transform(test)
bestrfmodel=cvmodel.bestModel
accuracy(rfpredict_test)
print('Best maxdepth:',bestrfmodel.getOrDefault('maxDepth'))
print('Best maxBins:',bestrfmodel.getOrDefault('maxBins'))
print('best numTrees:',bestrfmodel.getOrDefault('numTrees'))

#SVM on Relief Features
sv=LinearSVC(labelCol='target',featuresCol='scaledrelieffeatures')
svmodel=sv.fit(train)
svpredict_train=svmodel.transform(train)
svpredict_test=svmodel.transform(test)
accuracy(svpredict_test)
```

```
print('regParam:',svmodel.getOrDefault('regParam'))
print('maxIter:',svmodel.getOrDefault('maxIter'))

evaluator=BinaryClassificationEvaluator(rawPredictionCol='rawPrediction',labelCol='tar
get')
paramgrid=ParamGridBuilder().addGrid(sv.regParam,[0.01,0.02,0.03,0.1,0.2,0.3]).addGri
d(sv.maxIter,[10,20,25,30,35,40,45,50]).build()
cv=CrossValidator(estimator=sv,estimatorParamMaps=paramgrid,evaluator=evaluator,nu
mFolds=10)
cvmodel=cv.fit(train)
svpredict_train=cvmodel.transform(train)
svpredict_test=cvmodel.transform(test)
bestsvmodel=cvmodel.bestModel
accuracy(svpredict_test)
print('regParam:',bestsvmodel.getOrDefault('regParam'))
print('maxIter:',bestsvmodel.getOrDefault('maxIter'))

#Streaming
import pykafka
import json
import tweepy
from tweepy import OAuthHandler
from tweepy import Stream
from tweepy.streaming import StreamListener

#TWITTER API CONFIGURATIONS
consumer_key ='Fas47a2nQuaMB39MT1l4VCjvt'
consumer_secret
='uP6BWcSu8GV18BLNYC4D1DZDf8rajuMKAZAKHPFbRAFY1PEWJr'
access_token ='1971566503-uJ7siWrpWzlQQQK1zaULroPwT6je09RyzGNg1hP'
access_secret = 'EAEfQHsLwU3DxfM45dzzBYTryNJ2ZokcxmsNmcGZ2PCps'

#TWITTER API AUTH
auth = OAuthHandler(consumer_key, consumer_secret)
auth.set_access_token(access_token, access_secret)

api = tweepy.API(auth)

#Twitter Stream Listener
class KafkaPushListener(StreamListener):
    def __init__(self):
            #localhost:9092 = Default Zookeeper Producer Host and Port Adresses
            self.client = pykafka.KafkaClient("localhost:9092")
```

```
            #Get Producer that has topic name is Twitter
            self.producer = self.client.topics[bytes("hrtdis", "ascii")].get_producer()

    def on_data(self, data):
            #Producer produces data for consumer
            #Data comes from Twitter
            self.producer.produce(bytes(data, "ascii"))
            return True

    def on_error(self, status):
            print(status)
            return True

#Twitter Stream Config
twitter_stream = Stream(auth, KafkaPushListener())
print(twitter_stream)
#Produce Data that has hrtdis hashtag (Tweets)
twitter_stream.filter(track=['hrtdis'])

#Online Prediction
pip install findspark
import findspark
findspark.init()
from pyspark.ml.classification import RandomForestClassificationModel
from pyspark.ml.feature import MinMaxScaler
from pyspark.ml.feature import VectorAssembler
from pyspark.sql.types import *
from pyspark.sql import SparkSession
from pyspark.context import SparkContext
from pyspark.streaming import StreamingContext
from pyspark.streaming.kafka import KafkaUtils
import json
if __name__ == "__main__":
   sc = SparkContext(appName="trail")
   sc.setLogLevel("WARN")
   ss=SparkSession(sc)
   ssc = StreamingContext(sc,360)
   modelpath="D:\model"
   rfmodel=RandomForestClassificationModel.load(modelpath)
    #localhost:2181 = Default Zookeeper Consumer Address
   kafkaStream = KafkaUtils.createStream(ssc, 'localhost:2181', 'spark-streaming',
{'hrtdis':1})
```

```
    parsed = kafkaStream.map(lambda v: json.loads(v[1]))
    txt=parsed.map(lambda x: x['text']).map(lambda x: x.split())
    def f(x):
        try:

df=ss.createDataFrame(x,['h1','h2','AGE','SEX','CPT','RBP','SCH','FBS','RES','MHR','EI
A','OPK','PES','VCA','THA'])
            df=df.drop('h1','h2')
            df=df.withColumn('AGE',df['AGE'].cast(DoubleType()))
            df=df.withColumn('SEX',df['SEX'].cast(DoubleType()))
            df=df.withColumn('CPT',df['CPT'].cast(DoubleType()))
            df=df.withColumn('RBP',df['RBP'].cast(DoubleType()))
            df=df.withColumn('SCH',df['SCH'].cast(DoubleType()))
            df=df.withColumn('FBS',df['FBS'].cast(DoubleType()))
            df=df.withColumn('RES',df['RES'].cast(DoubleType()))
            df=df.withColumn('MHR',df['MHR'].cast(DoubleType()))
            df=df.withColumn('EIA',df['EIA'].cast(DoubleType()))
            df=df.withColumn('OPK',df['OPK'].cast(DoubleType()))
            df=df.withColumn('PES',df['PES'].cast(DoubleType()))
            df=df.withColumn('VCA',df['VCA'].cast(DoubleType()))
            df=df.withColumn('THA',df['THA'].cast(DoubleType()))
            df.show()
            cols=df.columns
            assembler=VectorAssembler(inputCols=cols,outputCol='featuresvector')
            df=assembler.transform(df)
            df.select('featuresvector').show()
            scaler=MinMaxScaler(inputCol='featuresvector',outputCol='scaledfeatures')
            scalermodel=scaler.fit(df)
            df=scalermodel.transform(df)
            df.select("scaledfeatures").show()
            df=rfmodel.transform(df)
            df.select('prediction').show()
        except:
            print('no patient data available')
    txt.foreachRDD(f)
    ssc.start()
    ssc.awaitTermination()
```

4. SNAPSHOTS

4.1 Univariate Feature Selection

The results of Univariate Feature Selection algorithm are depicted in Fig. 4.1 below, where the top seven features are marked black.

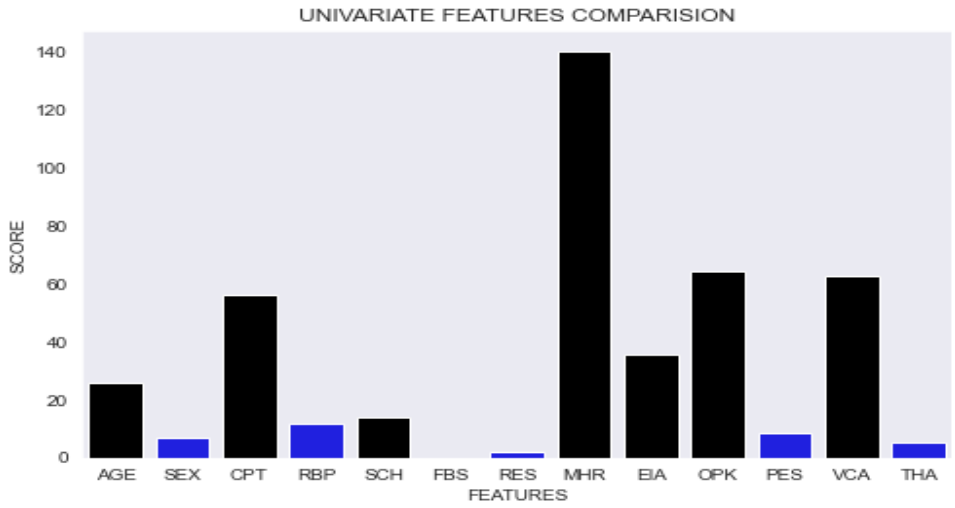

Fig. 4.1 Univariate Feature Selection Scores

The important features selected by Univariate Features selection are shown in Table 4.1 below:

Feature code	**Univariate score**
MHR	139.96
OPK	64.43
VCA	62.81
CPT	56.25
EIA	35.61
AGE	25.60
SCH	13.85

Table 4.1 Selected Univariate Features with score

4.2 Relief Feature Selection

The results of Relief Feature Selection algorithm are depicted in Fig. 4.2 below, where the top seven features are marked black.

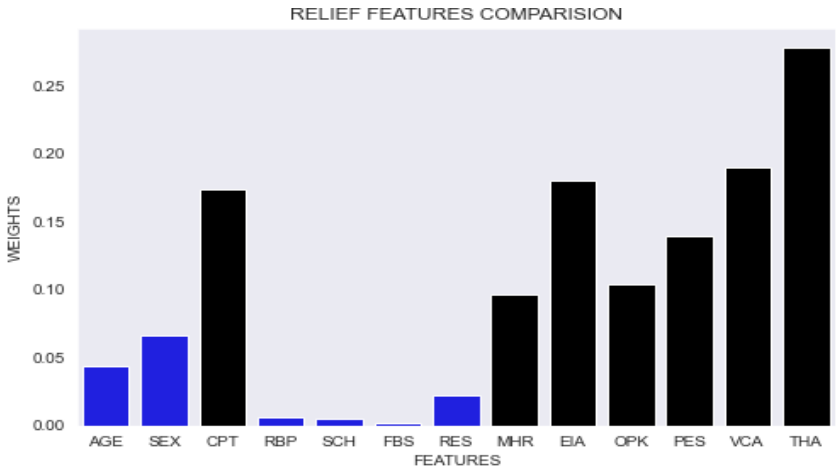

Fig. 4.2 Relief Feature Selection Ranking

The important features selected by Relief Features selection are shown in Table 4.2 below:

Feature code	**Relief ranking**
THA	0.278
VCA	0.186
EIA	0.179
CPT	0.173
PES	0.139
OPK	0.104
MHR	0.096

Table 4.2 Selected Relief Features with ranking

4.3 Accuracy Comparison

Fig. 4.3 depicts the comparison of accuracy between LR, DT, RF, and SVM Classifiers on Full Features. RF has achieved the highest accuracy of 89.9%.

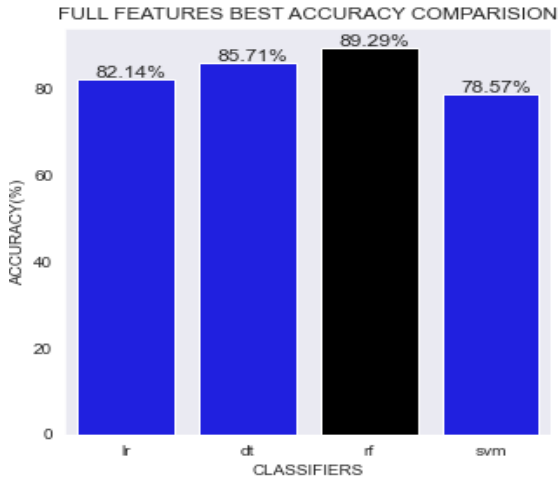

Fig. 4.3 Accuracy Comparison of Full Features

Fig. 4.4 depicts the comparison of accuracy between LR, DT, RF, and SVM Classifiers on Univariate Features. SVM has achieved the highest accuracy of 84.21%.

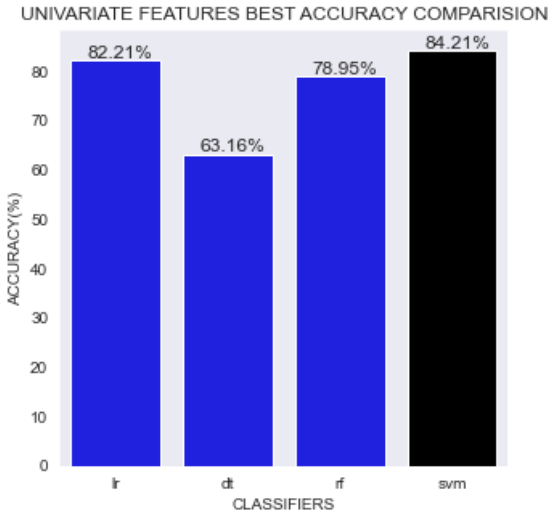

Fig. 4.4 Accuracy Comparison of Univariate Features

Fig. 4.5 depicts the comparison of accuracy between LR, DT, RF, and SVM Classifiers on Relief Features. LR, DT, and SVM have achieved the highest accuracy of 78.95%.

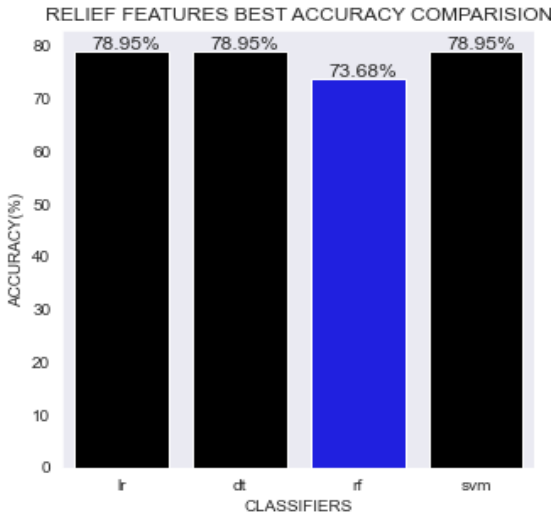

Fig. 4.5 Accuracy Comparison of Relief Features

4.4 Best Parameters Values

The best parameters values for Classifiers on Full Features are shown in Table 4.3 below:

Model	**Best Parameters- Full Features**
LR	regParam: 0.2, maxIter: 30
DT	impurity: entropy, maxDepth: 5, maxBins: 10
RF	maxDepth: 3, maxBins: 15, numTrees: 45
SVM	regParam: 0.1, maxIter: 35, kernel: Linear

Table 4.3 Best Parameters values- Full Features

The best parameters values for Classifiers on Univariate Features are shown in Table 4.4 below:

Model	Best Parameters- Univariate Features
LR	regParam: 0.2, maxIter: 10
DT	impurity: entropy, maxDepth: 4, maxBins: 35
RF	maxDepth: 4, maxBins: 10, numTrees: 35
SVM	regParam: 0, maxIter: 100, kernel: Linear

Table 4.4 Best Parameters values- Univariate Features

The best parameters values for Classifiers on Univariate Features are shown in Table 4.5 below:

Model	Best Parameters- Relief Features
LR	regParam: 0.3, maxIter: 10
DT	impurity: entropy, maxDepth: 5, maxBins: 10
RF	maxDepth: 5, maxBins: 35, numTrees: 20
SVM	regParam: 0.2, maxIter: 10, kernel: Linear

Table 4.5 Best Parameters values- Relief Features

4.5 Online Prediction Results

The Fig. 4.6 shows the results obtained by online prediction using the RF model on real-time twitter data.

S. No.	Tweet	Prediction
1	"* hrtdis 37.0 1.0 4.0 194.0 292.0 0.0 2.0 144.0 0.0 0.0 1.0 1.0 2.0"	1.0
2	"* hrtdis 40.0 1.0 1.0 114.0 282.0 1.0 2.0 114.0 0.0 2.0 1.0 1.0 7.0"	0.0
3	"* hrtdis 67.0 1.0 2.0 138.0 255.0 0.0 2.0 152.0 1.0 0.0 1.6 0.0 3.0"	1.0
4	"* hrtdis 57.0 1.0 3.0 154.0 196.0 0.0 2.0 288.0 0.0 0.0 1.0 1.0 7.0"	1.0
5	"* hrtdis 44.0 0.0 1.0 135.0 262.0 1.0 2.0 184.0 0.0 2.0 0.0 1.0 3.0"	0.0
6	"* hrtdis 61.0 1.0 4.0 148.0 203.0 0.0 0.0 161.0 0.0 0.0 1.0 1.0 7.0"	1.0
7	"* hrtdis 50.0 1.0 3.0 129.0 196.0 0.0 0.0 163.0 0.0 0.0 1.0 0.0 3.0"	1.0
8	"* hrtdis 65.0 0.0 3.0 111.0 167.0 1.0 2.0 135.0 0.0 2.0 1.0 1.0 7.0"	0.0
9	"* hrtdis 71.0 1.0 4.0 112.0 149.0 0.0 0.0 125.0 0.0 1.6 2.0 0.0 3.0"	1.0
10	"* hrtdis 65.0 0.0 3.0 171.0 167.0 1.0 2.0 155.0 0.0 0.0 1.0 1.0 3.0"	0.0

Fig. 4.6 Online Prediction Results

5. CONCLUSIONS AND FUTURE PLANS

A system developed on Apache Spark affiliated with Apache Kafka was presented in this paper. This system comprises three components: 1. Building Offline Model,2. Streaming Process Pipeline, and 3. Online Prediction. In Building Offline Model, the heart dataset was utilized and ML models LR, DT, RF, and SVM have been developed on Full Features besides Selected Features and accuracy was enhanced by using hyperparameter tuning and K-Fold Cross-Validation techniques. By selecting the important features using Univariate and Relief Feature Selection algorithms, the accuracy wasn't enhanced compared to the Full Features in which RF achieved the best accuracy of 89.9%. RF was selected as the best model because in Online Prediction we will use Full Features extracted from the tweets. The Streaming Process Pipeline was created by integrating Apache Spark and Apache Kafka. Apache Kafka will retrieve the tweets with header “hrtdis” and stores it in Kafka Topic. Apache Spark will read the data in form of streams from the created Kafka Topic and process the data and converts it into vector. The Online Prediction component will receive the vector from Apache Spark and prediction will be done by using the RF model. The RF model showed better accuracy than other competitive works.

In future, other methodologies like Deep Learning algorithms can be incorporated and enhance the accuracy further to get more accurate predictions.

6. REFERENCES

[1] Hager Ahmed, Eman M.G. Younis, Abdeltawab Hendawi, Abdelmgeid A. Ali, Heart disease identification from patients' social posts, machine learning solution on Spark, *Future Generation Computer Systems, Volume 111, 2020, Pages 714-722, ISSN 0167-739X*.

[2] Cleveland, S. Hungary**,** The VA Long Beach, 2019. Heart disease data set. https://archive.ics.uci.edu/ml/datasets/heart+Disease.

[3] G. Manogaran, R. Varatharajan, M. Priyan, Hybrid recommendation system for heart disease diagnosis based on multiple kernel learning with adaptive neuro-fuzzy inference system, *Multimedia Tools Appl. 77 (4) (2018) 4379–4399.*

[4] S. Nazari, M. Fallah, H. Kazemipoor, A. Salehipour, A fuzzy inference-fuzzy analytic hierarchy process-based clinical decision support system for diagnosis of heart diseases, *Expert Syst. Appl. 95 (2018) 261–271*.

[5] V. Jayaraman, H.P. Sultana, Artificial gravitational cuckoo search algorithm along with particle bee optimized associative memory neural network for feature selection in heart disease classification, J. *Ambient Intell. Hum. Comput. (2019) 1–10.*

[6] C.B. Gokulnath, S. Shantharajah, An optimized feature selection based on genetic approach and support vector machine for heart disease, *Cluster Comput. (2018) 1–11*.

7. PLAGIARISM REPORT

ORIGINALITY REPORT

3%	%	3%	%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	Hager Ahmed, Eman M.G. Younis, Abdeltawab Hendawi, Abdelmgeid A. Ali. "Heart disease identification from patients' social posts, machine learning solution on Spark", Future Generation Computer Systems, 2020 Publication	2%
2	Jaishri Wankhede, Magesh Kumar, Palaniappan Sambandam. "Efficient heart disease prediction-based on optimal feature selection using DFCSS and classification by improved Elman-SFO", IET Systems Biology, 2020 Publication	<1%

Exclude quotes	Off	Exclude matches	Off
Exclude bibliography	Off		

Future Generation Computer Systems xxx (xxxx) xxx

Contents lists available at ScienceDirect

Future Generation Computer Systems

journal homepage: www.elsevier.com/locate/fgcs

Heart disease identification from patients' social posts, machine learning solution on Spark

Hager Ahmed [a,*], Eman M.G. Younis [a], Abdeltawab Hendawi [b,c], Abdelmgeid A. Ali [d]

[a] Information Systems Department, Faculty of Computers and Information, Minia University, Egypt
[b] Department of Computer Science and Statistics, University of Rhode Island, USA
[c] Faculty of Computers and Artificial Intelligence, Cairo University, Egypt
[d] Computer Science Department, Faculty of Computers and Information, Minia University, Egypt

ARTICLE INFO

Article history:
Received 13 June 2019
Received in revised form 1 September 2019
Accepted 27 September 2019
Available online xxxx

Keywords:
Heart disease prediction
Machine learning
Streaming data
Apache Spark
Apache Kafka

ABSTRACT

Heart disease is one of the first causes of death worldwide. This paper presents a real-time system for predicting heart disease from medical data streams that describe a patient's current health status. The main goal of the proposed system is to find the optimal machine learning algorithm that achieves high accuracy for heart disease prediction. Two types of features selection algorithms, univariate feature selection and Relief, are used to select important features from the dataset. We compared four types of machine learning algorithms; Decision Tree, Support Vector Machine, Random Forest Classifier, and Logistic Regression Classifier with the selected features as well as full features. We apply hyperparameter tuning and cross-validation with machine learning to enhance accuracy. One core merit of the proposed system is able to handle Twitter data streams that contain patients' data efficiently. This is done by integrating Apache Kafka with Apache Spark as the underlying infrastructure of the system. The results show the random forest classifier outperforms the other models by achieving the highest accuracy at 94.9%.

1. Introduction

Heart diseases are on top of the fatal diseases list. They are considered a prominent cause of death globally. According to the World Health Organization statistics, in 2016, heart diseases led to about 17.9 million deaths [1].

In the USA, heart diseases such as Coronary Heart Disease, Hypertension, and Stroke are the first fatal cause of death. Coronary Heart Disease alone contributes to 1 in every 7 losses of life in the US, with around 366,800 death cases annually. The estimation of heart attacks in the USA is about 7.9 million representing approximately 3% of US adults' heart attack cases. In the same country, 114,023 people died due to a heart attack in 2015 [2].

Therefore, there is a persistent need for a highly-accurate system that serves as an analysis tool to discover hidden patterns of heart diseases in medical data and the prediction of heart attacks before they happen. This is hoped will lead to better management of heart attacks.

The classification algorithms have been used increasingly to predict heart diseases. For example, the back-propagation neural network (BPNN) [3] and logistic regression (LR) [3] were applied for the prediction of heart diseases using Cleveland dataset. Moreover, Novel Multi-Layer Pi-Sigma Neuron Model (MLPSNM) was proposed to predict heart disease [4]. In addition, there are many algorithms like Naive Bayes [5], Decision Tree [5] and k-Nearest Neighbor (KNN) [5] which were used to predict heart diseases.

Recently, there have been many hybrid techniques proposed for heart disease diagnosis and prediction. The hybrid method consists of two main stages. The first stage is feature selection, which is used to select a subset of features. In the second stage, the selected subset of features was used as the training set for building the classification models [6]. For example, in [7] heart disease diagnosis system was developed by a combination of rough sets based attribute reduction with interval type-2 fuzzy logic. Also, the hybrid technique using decision tree and an artificial neural network was developed for the prediction of heart diseases [8].

However, the enormous and overwhelming volumes of the historical data and the continuous flow of streaming data that are generated in healthcare services have become an unprecedented challenging task to process, store, and analyze using traditional database storage and machine learning methods. For processing the data in real-time, many companies and researchers have used big data frameworks. The Apache Spark, Apache Flink, and Apache Storm have been applied for real-time data processing. With

* Corresponding author.
E-mail addresses: hagar.salehPG@fci.s-mu.edu.eg (H. Ahmed), eman.younas@mu.edu.eg (E.M.G. Younis), hendawi@uri.edu, a.hendawi@fci-cu.edu.eg (A. Hendawi), dean.fci@mu.edu.eg (A.A. Ali).

https://doi.org/10.1016/j.future.2019.09.056
0167-739X/

the new processing frameworks, it has become more efficient to analyze the streaming data. Therefore, some researchers have done a shift towards the use of Big Data platforms and distributed machine learning [9–12].

Previous studies in the field of heart disease prediction have focused only on predicting heart diseases based on previously stored data, and traditional machine learning algorithms to solve this problem. These studies cannot predict heart diseases with real-time streaming data. Regarding the data sets, the previous studies have not considered data of social media platforms as a source to solve the problem of heart disease prediction. Still, an exception to this is the study presented by Nair, L.R., [13] who developed a system using decision tree algorithms to predict heart disease in real-time from Twitter. This contribution relied on using only one algorithm in building the model which is the decision tree, without using feature selection nor any other type of machine learning algorithms with the model to achieve high accuracy.

Twitter is one of the social media platforms which are rich in medical information that is used increasingly for various health and medicinal purposes including sharing information about diabetes [14], identifying the potential adverse drug [15], analyzing breast cancer [16] and others [17]. In addition, plenty of researchers have used Twitter to conduct their research studies and their experiments such as [18,19]. The main advantages of using the Twitter platform is the ability to evaluate the system in real-time and to make the system scalable through receiving a large number of data from Twitter in real-time, as well as measuring the ability of the system for predicting whether the tweet contains indications of heart disease or not.

In this paper, the problem of predicting heart diseases using a set of streaming data collected from users' social network activities is addressed. The main goal of the work is to develop a model using the Cleveland heart disease dataset 2016 to achieve high accuracy and then use this model to work in real-time to classify the tweet as containing heart disease indication or not. The real-time heart disease prediction system consists of three main components: Offline Model Building, Streaming Processing Pipeline, and Online Prediction. The system was developed based on Big Data frameworks like Apache Spark [20] and Apache Kafka [21]. Also, feature selection algorithms Relief and Univariate feature selection were employed to select the important set of features from the database. Moreover, four machine learning algorithms which are Decision Tree (DT), Support Vector Machine (SVM), Random Forest Classifier (RF) and Logistic Regression Classifier (LR) have been applied on all the features and a subset of the selected features; and we used k-fold (cross-validation) with hyperparameter tuning to improve the accuracy. The main contributions of this work are:

- Developing a real-time system that can determine and extract knowledge related to heart diseases from user's streaming tweets to predict whether the person is prone to have heart disease or not.
- Applying two feature selection algorithms to select the most important features from the data set. In addition, an experimental evaluation was conducted for comparing different machine learning classification techniques such as DT, SVM, RF, and LR in terms of their accuracy for the full features and the selected set of features. Then, the best model that achieved the highest accuracy was implemented to predict the possibility of heart disease problems from streams of tweets.
- Applying RF, which gave the best accuracy of 94.9% for full features of Cleveland heart disease dataset. This accuracy achieved is 11% higher than the decision tree algorithm applied in [13].

The rest of the paper is structured as follows. Section 2 provides a literature review of the previous studies. Section 3 explains the background of big data tools. Section 4 explains the structures of the dataset. Section 5 describes the proposed system that predicts heart diseases. Section 6 presents the experimental results. Finally, Section 7 presents conclusions.

2. Literature review

Heart diseases have been recognized as one of the leading causes of death. Numerous research studies investigated the use of predictive models to predict indicators of suffering from heart diseases through health data in the literature. The recent advances in machine learning tools and algorithms have led to boosting the research in the development of methods and techniques for heart disease diagnosis. Many techniques have been investigated for solving this problem, such as classification, clustering, and many more.

Many hybrid models were proposed in the previous studies. For example, Nazari et al. [22] introduced a system using the Fuzzy Analytic Hierarchy Process (AHP) and Fuzzy Inference. A data set from a hospital in Tehran was used to train and test this system. The Fuzzy AHP was employed for the purpose of calculating the weights of various criteria that contribute to the development of heart diseases, while the Fuzzy Inference System was used for assessing and evaluating the potential of developing heart diseases in a patient. Manogaran et al. [23] proposed a combination system of multiple kernel learning and neuro-fuzzy inference for heart disease diagnosis. The system was tested on the metabolic reaction network dataset and achieved high sensitivity at 98%, and high specificity at 99%.

Feature selection techniques were used to reduce the number of features to simplify the model. In this context, many researchers have used various feature selection techniques with machine learning models to diagnose heart diseases. In [24], SVM with a genetic algorithm (GA) was used to find the most important features for classifying heart diseases. A feature selection technique is the GA, while the SVM is a classification algorithm. Using the Cleveland heart disease database, the accuracy of SVM-GA increased from 83.70% for 13 features to 88.34% after using the feature selection to reduce the number of features for creating the model. Similarly, in [25] Artificial Gravitational Cuckoo Search Algorithm Along with Particle Bee Optimized Associative Memory Neural Network approach was used as a feature selection method.

3. Background

Apache Spark and Apache Kafka were used to develop the proposed system. This section provides a brief background and advantages of big data tools.

3.1. Apache Spark

Apache Spark [20] is an open-source distributed framework for data analytics that provides fault tolerance and process data in real-time. It provides in-memory processing that allows data and intermediate results to be kept in memory, thus avoiding input–output delay of switching data back and forth the hard disk [26]. Spark can work with structured data like CSV files and unstructured data such as JSON format. It provides high-level APIs such as Spark Streaming and MLlib. Spark Streaming, a streaming library of Spark, reads streaming tweets from Kafka topic and processes data in real-time. MLlib, the machine learning library of Spark, implements machine learning classification algorithms, hyperparameter tuning, and cross-validation.

Please cite this article as: H. Ahmed, E.M.G. Younis, A. Hendawi et al., Heart disease identification from patients' social posts, machine learning solution on Spark, Future Generation Computer Systems (2019), https://doi.org/10.1016/j.future.2019.09.056.

Table 1
Features information and description of Cleveland heart disease dataset 2016.

S.no	Feature name	Code
1	Age	AGE
2	Sex	SEX
3	Type of chest pain	CPT
4	Resting blood pressure	RBP
5	Serum cholesterol	SCH
6	Fasting blood sugar	FBS
7	Resting electrocardiographic results	RES
8	Maximum heart rate achieved	MHR
9	Exercise-induced angina	EIA
10	Old peak = ST depression induced by exercise relative to rest	OPK
11	Slope of the peak exercise ST segment	PES
12	Number of major vessels (0–3) colored by fluoroscopy	VCA
13	Thallium scan	THA

3.2. Apache Kafka

Apache Kafka was used for building real-time data pipelines and streaming apps. The main advantages of Kafka are receiving a large amount of data in real-time with low latency while remaining fault-tolerant with more scalability. Kafka [21] is a streaming platform that is used to read data as streaming from Twitter. Kafka can read and write a large number of data streams. It can write scalable stream processing applications that react to events in real-time. It stores data streams safely in a distributed, replicated, and fault-tolerant cluster. The main libraries are Producer API and Consumer API. Producer API allows an application to send a stream of records to Kafka topics. The Consumer API allows an application to subscribe to Kafka topics and process the stream of records.

4. Dataset

In this section, we present the details of the dataset used for building the heart disease prediction model. Also, we describe the structure of the dataset used to evaluate the model in real-time.

4.1. The heart dataset

This section describes the dataset used for building the offline model. Cleveland heart disease dataset 2016 [27] has been used for training and testing the machine learning algorithms for predicting heart disease. This dataset consists of 13 independent variables as features and one dependent variable as the class label that is used to predict heart disease. Table 1 describes the full information about the features. The class label has five values, which are 0 class label representing the absence of heart disease; while values 1, 2, 3, and 4 represent the presence of some heart problem. In this paper, our work is a binary classification problem to either 1 for the person who has heart disease, or 0 for the person who does not have heart disease. Therefore, we have replaced the values 2,3,4,5 with 1 because the meaning here is that the person has heart disease.

4.2. Twitter real-time data collection

Twitter is widely used, as well-known, as a social platform for people's interaction, for posting content, sharing, and commenting where people discuss various topics including health. Therefore, it is considered a rich source of real-time data related to health. The streams of tweets are received from Twitter to evaluate the proposed system in real-time. Moreover, It measures the efficiency of the proposed system to work in real-time and its ability to predict whether the tweet includes heart disease or not. For reading tweet streams, the system needs to create an authorization connection with Twitter using Twitter Streaming API [28]. Twitter uses OAuth, which is the authentication protocol proposed to authorize applications to access services on the user's behalf without sharing the password. For this, we create an account from the Twitter app, which provides the consumer key and the secret consumer key, access token, and access token secret for authorized access of tweet streams. After, the system establishes an authorization connection; it retrieves tweets by the header word "hrtdis". Fig. 1 presents an example of the tweet is read by the system. This tweet includes a sequence of attributes value like AGE, SEX, CPT, RBP, SCH, FBS, RES, MHR, EIA, OPK, PES, VCA and THA, as the same order in the training dataset.

```
hrttest3;2015-09-29 06:21;0;0;"* hrtdis 52.0 1.0
4.0 128.0 255.0 0.0 0.0 161.0 1.0 0.0 1.0 1.0 7.0"
```

Fig. 1. Example of the collected tweet.

5. The heart disease prediction system

This section presents an explanation of the overall architecture of the system.

5.1. The architecture of heart disease prediction system

The online predictive heart disease system was implemented on Apache Spark. The architecture of the proposed system includes three main components, namely, Building Offline Model, Stream Processing Pipeline, and Online Prediction, as shown in Fig. 2. Each component is explained in detail in the following sections.

5.1.1. Building the offline model

The main goal of this component is to develop the machine learning model that achieves the highest possible accuracy throughout experimentation. Many classification algorithms were used DT, SVM, RF, and LR to build the offline-model. In addition, feature selection algorithms Relief and Univariate were used to select the important features in the database, which must be present to make the system able to predict heart disease correctly. Fig. 3 describes the architecture of the first component of the system, building the offline model. This component is composed of four stages as follows: (1) data pre-processing, (2) feature selection, (3) machine learning classifiers, and (4) evaluating machine learning algorithms.

(A) Data pre-processing is of the essential steps for effectively representing the data for the machine learning algorithm that is required to be trained and tested in effect. For practical use in the classifiers, pre-processing techniques including missing values removal and MinMax Scalar. MinMax Scalar works on shifting the data so that all the feature's values are between 0 and 1. All of these data pre-processing techniques have been applied in the first step.

(B) Feature selection algorithms
The main advantages of using feature selection algorithms are determining the important features in the database, and we have used two types of algorithms which are univariate feature selection and Relief to specify these important features which must be present to make the system able of predicting heart disease correctly, as well as knowing the features that if absent will not affect the predictive ability of the system.

Please cite this article as: H. Ahmed, E.M.G. Younis, A. Hendawi et al., Heart disease identification from patients' social posts, machine learning solution on Spark, Future Generation Computer Systems (2019), https://doi.org/10.1016/j.future.2019.09.056.

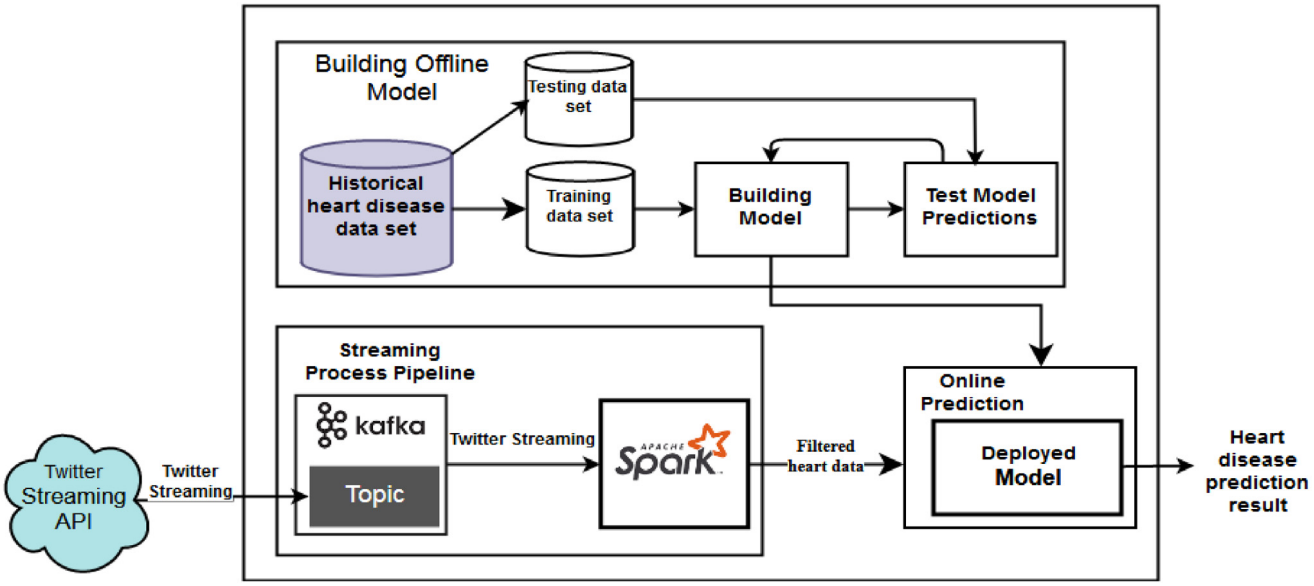

Fig. 2. The architecture of the heart disease prediction system.

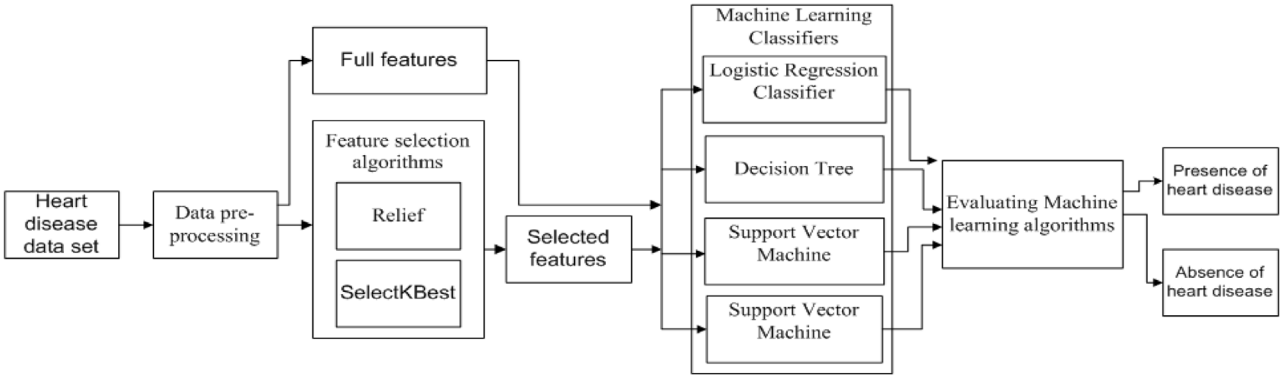

Fig. 3. The architecture of building offline model.

(I) Univariate feature selection uses a statistical test to select a feature subset that has the strongest relationship with a class label. We used SelectKBest [29] with chi-squared statistic test chi2(X, y) [30]. The chi-square test measures the dependence between stochastic variables. Therefore, utilizing this function helps to get rid of the features that are the most likely to be independent of class and, irrelevant for the classification. The obtained score is used to select the n_features, with the highest values for the chi-squared statistic from X.

(II) Relief feature selection algorithm is a feature selection algorithm [31], which assigns weights to all the features in the dataset, and these weights are updated with over time. The essential features haveing higher weights are used in the model. The remaining features with small weights are ignored.

(C) Machine Learning Classifiers
In this section, the classification algorithms used in the proposed system are explained.

(I) Logistic regression, a classification algorithm, is used for a binary classification problem. For a binary classification problem, LR is used to predict the value of the predictive variable y when y [0,1] is 0 a negative class, and 1 is a positive class [31].

(II) Support vector machine is a supervised machine learning algorithm which is used for both classification and regression problem. It has been used mostly for classification problems [6,32]. The goal of linear SVM is to find the best hyperplane that can separate the dataset into two classes. It divides the dataset into two classes 0 and 1 that are situated on both sides of the hyperplane.

(III) Decision tree is a supervised machine learning algorithm that is used to solve the classification problem. It is a tree structure. It includes a collocation of nodes, where each internal node represents features, each leaf node represents a class label, and each branch represents an outcome of the test [33].
This algorithm, in a repeated manner, splits the data set based on a criterion that maximizes the separation of the data, which results in a tree-like structure. The most common test used is information gain; this means that at every split, the decrease in entropy is maximized because of this split. The estimate of P(y|x) is the ratio of y class elements over all elements of the leaf node that contains data item x [33].

(IV) Random forest is an ensemble machine learning technique that can be used for classification problem [34]. The RF consists of a combination of tree

classifiers. Each tree classifier is created by a random vector sampled independently from the input vector, and each tree casts a unit vote for the most popular class to classify an input vector [35].

(D) Evaluating machine learning algorithms
K-Fold Cross-validation (CV) method and hyperparameter tuning were used to get the best accuracy for the models. In addition, the performance evaluation of the classification techniques was done through the average accuracy of 10-fold cross-validation.
Cross-validation, hyperparameter, and accuracy are described in details in the following:

(I) Hyperparameter tuning is utilized to pass various parameters into the model. Grid Search is the most used method for applying hyperparameter tuning. Firstly, the user defines a set of values for each hyperparameter. Then, the model tests all values for each hyperparameter and selects the best value that achieves the best accuracy.
(II) K-Fold Cross-Validation: the data set is split into k equal size of fold. k-1 groups are utilized for the training, and the remaining part is used to test the classifiers. This process is repeated until each fold of the 10 folds has been used as the testing set. Also, for each k, the accuracy of classifiers is calculated. Finally, the evaluation classifier is computed based on the average accuracy, which is calculated at the end of the 10-fold process. In our experiment, we used $k = 10$. In the 10-fold CV process, 90% of data were used for the training, and 10% of data were used for testing purposes.
(III) Accuracy is the ratio between correct results to a number of total predictions that define in the following:

$$Accuracy = \frac{Number\ of\ Correct\ Prediction\ Class}{Total\ Number\ of\ Prediction\ Class}$$

5.2. Streaming processing pipeline

This stage is considered one of the main stages used for evaluating the model in real-time. The system retrieves tweets by the header word "hrtdis". This tweet includes a sequence of attribute values like AGE, SEX, CPT, RBP, SCH, FBS, RES, MHR, EIA, OPK, PES, VCA, and THA as the same order in the training dataset. The stage consists of many steps where data is received from Twitter and sent to Kafka Topic. After that, Spark reads the streaming data from Kafka Topic. The collected data is text format. Therefore, some steps must be applied, including removing unimportant data and then extracting data, which is related to health. Later, it transforms this data into a vector and sends it in the same order as the feature vector in training set to the Random Forest Classifier, which was developed.

5.3. Online prediction

The developed model, which had achieved the best accuracy, receives the data extracted from the tweet in the form of a vector to predict whether this tweet contains indications of heart disease or not.

Table 2
Features selected by Univariate and their scores.

Feature code	Scores
MHR	188.32
OPK	72.644
VCA	66.441
CPT	62.598
EIA	38.914
AGE	23.287
SCH	23.936

6. Experimental results and discussion

6.1. Experimental setup

The proposed system based on Apache Spark was written in Python. In addition, we used different API libraries that are integrated with Spark. Spark's Mlib is used to implement classification algorithms. Kafka is a scalable platform that is used to receive streaming data from Twitter. Spark Streaming was used to read tweets streaming from a Kafka topic. We have used Python libraries to implement the feature selection algorithms.

The propped system was implemented on a Spark cluster which consists of one master node and two worker nodes. We used Ubuntu 14.04 virtual machines that run Java (VM) to build the cluster. The name node and worker nodes have 20 GB of RAM, seven cores, and 100 GB disk.

6.2. Results of applying feature selection algorithms

Our research paper focuses on two feature selection techniques, which are Univariate and Relief that were used to select the important features from the dataset. The results of the selected features are represented in detail in the following two sections.

6.2.1. Selected features by Univariate

Table 2 displays the important 7 features which have the highest scores and were determined by the Univariate feature selection algorithm. It can be noticed that MHR has the highest score at 188.32 with a big difference compared to the second feature, which is OPK at 72.644. The table also shows that there is no significant difference between VCA at 66.441 and CPT at 62.598. At the bottom comes both AGE and SCH with almost a similar score, 23.287 and 23.936, respectively. It is also important to note that MHR has 8 times higher score than the SCH. The term of all features score is shown in Fig. 4. We can notice that MHR and OPK have the highest scores, and they have a considerable influence on model training and testing. FBS and RES have the smallest score at 2 and 2.97 respectively. The figure also displays the scores of PES, THA, SEX, and RBP, which have scored lower than 20.

6.2.2. Selected features by Relief

Relief selects important features depending on ranking and choosing the highest-ranking features. The important 7 features which have high rank are given in Table 3. We can notice that THA has the highest rank at 0.275. The table also shows that there is no notable difference between EIA at 0.172 and CPT at 0.179. At the bottom, there are both MHR and OPK with almost a similar score, 0.106 and 0.108, respectively. Fig. 5 displays the ranking of all features. According to the figure, THA and EIA have the highest ranking; and they have a considerable influence on model training and testing. SCH and FBS have the smallest ranking at 0.005 and 0.002, respectively.

Please cite this article as: H. Ahmed, E.M.G. Younis, A. Hendawi et al., Heart disease identification from patients' social posts, machine learning solution on Spark, Future Generation Computer Systems (2019), https://doi.org/10.1016/j.future.2019.09.056.

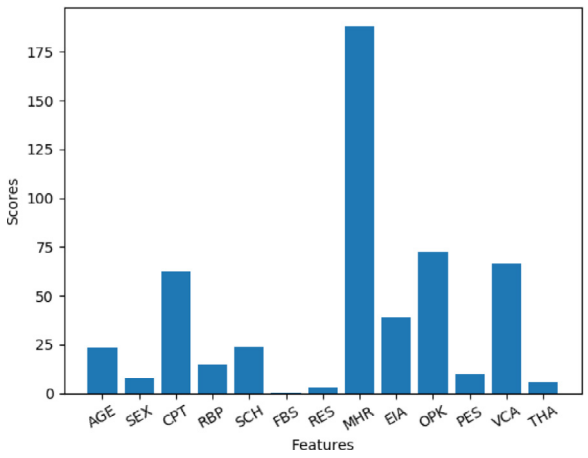

Fig. 4. The score of all features that were selected by Univariate.

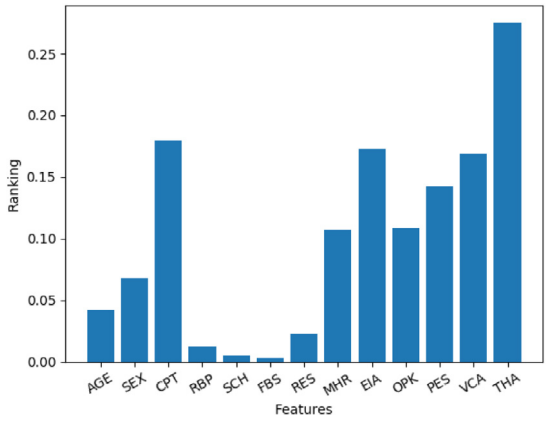

Fig. 5. The ranking of all features that were selected by Relief.

Table 3
The important features selected by Relief and their ranking.

Feature code	Ranking
THA	0.2753465346534657
EIA	0.17224422442244233
CPT	0.17953795379537962
VCA	0.16861386138613887
PES	0.14178217821782202
MHR	0.10674829314992579
OPK	0.10869796657085068

6.3. Results of applying the machine learning algorithms

The four classification techniques, which are SVM, RF, LR, and DT classifiers, were applied to the full set of features and only the selected features. The 10-fold cross-validation and hyperparameter tuning were applied with machine learning algorithms. For 10-fold cross-validation, the testing data used 10% while the training data used 90%. Moreover, the average accuracy of 10-fold is computed to evaluate models. Furthermore, some parameters were tuned into machine learning algorithms. For LR, two parameters were tuned: the maximum number of iterations (maxIter) and regularization parameter (regPram). Maximum depth of the tree (maxDepth) and max number of bins for discretizing continuous features (maxBins) and information gain (impurity) were tuned in DT. For SVM, three parameters were tuned: the maximum number of iterations (maxIter) and regularization parameter (regPram) and Kernal type. For RF, two parameters were

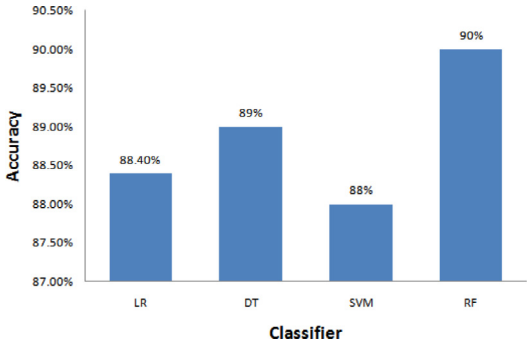

Fig. 6. The accuracy of different classifiers which applied on features that were selected by Univariate.

Table 4
The best values of parameters achieved the largest accuracy for features that were selected by Univariate.

Model	Parameters
LR	maxIter: 10 regPram: 0.01
DT	impurity: gini maxDepth: 3 macBins: 10
SVM	regParam:.02 maxIter:50 Kernal type: Liner
RF	maxDepth:2 maxBins:32 numTrees:20

Table 5
The best values of parameters which achieved high accuracy for features selected by Relief.

Model	Parameters
LR	regParam: .3 maxIter: 10
DT	impurity: gini maxDepth: 4 macBins: 10
SVM	regParam: .02 maxIter: 10 Kernal type: Liner
RF	maxDepth: 2 maxBins: 32 numTrees: 20

tuned: Maximum depth of the tree (maxDepth) and Max number of bins for discretizing continuous features (maxBins).

Moreover, for each classification algorithm, the values of the model's parameters were different. For example, in LR, the values of the maxIter and regPram parameters were 4 and 6, respectively. For DT, the values of the impurity, maxDepth and macBins parameters are 2, 6, and 4, respectively. For SVM, the values of the regParam and maxIter parameters are w and 4, respectively. In RF, the values of the numTrees, maxDepth, and maxBins parameters were 4, 10, and 3, respectively.

The results of K-Fold Cross-Validation for classifiers are represented in detail in the following section.

6.3.1. Accuracy using selected features selected by Univariate

The features MHR, OPK, VCA, CPT, EIA, AGE and SCH were selected from the database, and four types of machine learning classifiers were applied using these features. Table 4 shows the best values of the model's parameters, which contributed to registering the highest accuracy.

Fig. 6 shows a comparison of accuracy between four classifiers: RF, SVM, DT, and LR. It can be observed that RF has the highest accuracy at 90%, while SVM has the lowest accuracy at 88%. Similarly, DT and LR have recorded 89% and 88.40%, respectively.

6.3.2. Accuracy using selected features selected by Relief

THA, EIA, CPT, VCA, PES, MHR, and OPK were selected from the database, and four types of machine learning classifiers were applied to these features. Table 5 shows the best value of the model's parameters that are passed to classifiers indicating their essential role to achieve high accuracy.

Fig. 7 shows the average accuracy with the 10-fold CV. DT achieved the highest accuracy at 92.82%. The second important

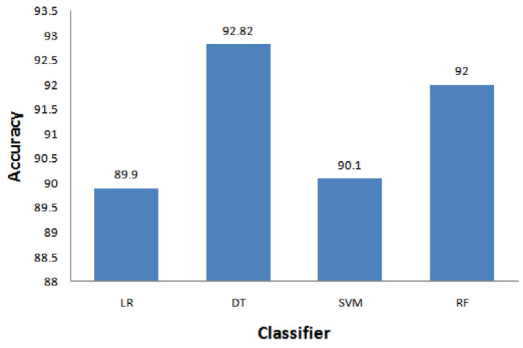

Fig. 7. The accuracy of different classifiers which applied on features selected by Relief.

Fig. 8. The average of accuracy with the 10-fold CV for full features.

Table 6
The best values of parameters for full features.

Model	Parameters
LR	regParam: .4 maxIter: 50
DT	impurity: entropy' maxDepth: 3 macBins: 10
SVM	regParam: .02 maxIter: 10 Kernal type: Liner
RF	maxDepth: 2 maxBins: 10 numTrees: 40

classifier was RF which recorded an accuracy of 92%. Approximately similar accuracy was observed for LR and SVM with percentages of 89.9% and 90.1% respectively.

6.3.3. Accuracy using full features

Table 6 shows the best values of model's parameters that were passed to the classifiers and their important role to achieve high accuracy.

In Fig. 8, the RF achieved the best accuracy at 94.9%. DT registered the lowest accuracy at 89.88%. LR made 92.12% of accuracy, while the SVM recorded 91.95% of accuracy.

6.4. Discussion

We applied two types of feature selection algorithms: Relief and Univariate that were used to select the most important features from the dataset. The results show that DT scored the highest accuracy at 92.2% with the features that were selected by the Univariate feature selection algorithm. The RF obtained the highest accuracy at 90% with features chosen by the Relief algorithm. In addition, applying RF gave the best accuracy of 94.9% using the full features. The achieved accuracy is 11% higher than the decision tree algorithm applied in [13]. Therefore, we used the RF with the full features to predict heart diseases in real-time from tweet streams.

6.5. Evaluating the proposed system using streams of tweets

In this stage, RF is used to evaluate the system in real-time with the tweet, which includes full features. In this experiment, the real streaming data from Twitter were used to assess the ability of the proposed system to predict whether a tweet contains heart disease or not in real-time. The total number of streaming tweets is 1000 generated by some of the twitter users. The results showed that there were 600 tweets that indicate the existence of a heart disease problem, while 400 tweets did not contain heart disease indications. Furthermore, Table 7 shows the structure of some tweets and the prediction label. The table also shows that five of the displayed tweets have heart disease, and five of them do not have heart disease.

7. Conclusions

In this paper, we presented a system for real-time heart disease prediction that was developed based on Apache Spark and Apache Kafka. Our real-time system consists of three components, namely Offline Model Building, Stream Processing Pipeline, and Online Prediction. Offline Model Building is a machine learning model that can achieve high accuracy. In this component, we analyze the features in the dataset and select the optimal set of features based on two feature selection algorithms, which are Univariate feature selection and Relief. In addition, Different machine learning classification algorithms, namely, SVM, DT, RF, and LR were applied to the full features and selected features. Furthermore, k-fold cross-validation and hyperparameters tuning were employed to improve accuracy. In the stream Processing Pipeline component, streams of the tweet are retrieved from Twitter using the header word "hrtdis" and sent to Kafka topic; then Spark stream reads data from Kafka topic and extracts the health attributes from it and then forwards it to the Online Prediction component. Online Prediction receives attributes as a vector of features in the same order as the sequence in the training dataset; then it applies the developed machine learning model to predict whether the tweet contains indications of heart disease or not. The results proved that the accuracy of RF with full features had improved the accuracy by 11% compared to competitive work.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Ethical approval

All procedures performed in studies involving human participants were in accordance with the ethical standards of the institutional and/or national research committee and with the 1964 Declaration of Helsinki and its later amendments or comparable ethical standards.

Informed consent

Informed consent was obtained from all individual participants included in the study.

Please cite this article as: H. Ahmed, E.M.G. Younis, A. Hendawi et al., Heart disease identification from patients' social posts, machine learning solution on Spark, Future Generation Computer Systems (2019), https://doi.org/10.1016/j.future.2019.09.056.

Table 7
The result of heart disease prediction from some of the tweets.

Sequence	Tweet	Prediction label
1	"* hrtdis 61.0 1.0 4.0 140.0 207.0 0.0 2.0 138.0 1.0 1.9 1.0 1.0 7.0"	0
2	"* hrtdis 57.0 1.0 2.0 154.0 232.0 0.0 2.0 164.0 0.0 0.0 1.0 1.0 3.0"	1
3	"* hrtdis 61.0 1.0 4.0 148.0 203.0 0.0 0.0 161.0 0.0 0.0 1.0 1.0 7.0";	1
4	"* hrtdis 71.0 0.0 4.0 112.0 149.0 0.0 0.0 125.0 0.0 1.6 2.0 0.0 3.0"	1
5	"* hrtdis 50.0 1.0 3.0 129.0 196.0 0.0 0.0 163.0 0.0 0.0 1.0 0.0 3.0"	1
6	"* hrtdis 57.0 0.0 2.0 130.0 236.0 0.0 2.0 174.0 0.0 0.0 2.0 1.0 3.0"	1
7	"* hrtdis 52.0 1.0 4.0 128.0 255.0 0.0 0.0 161.0 1.0 0.0 1.0 1.0 7.0"	0
8	"* hrtdis 65.0 1.0 1.0 138.0 282.0 1.0 2.0 174.0 0.0 1.4 2.0 1.0 3.0"	0
9	"* hrtdis 40.0 1.0 4.0 110.0 167.0 0.0 2.0 114.0 1.0 2.0 2.0 0.0 7.0"	0
10	"* hrtdis 54.0 1.0 3.0 125.0 273.0 0.0 2.0 152.0 0.0 0.5 3.0 1.0 3.0"	0

References

[1] W.H. Organization, World Health Organization, http://www.who.int/cardiovascular_diseases/en, 2019.
[2] A.H. Association, et al., Heart disease and stroke statistics-2003 update, http://www.americanheart.org/downloadable/heart/1059017971148200 3HDSStatsBookREV7-03.pdf, 2002.
[3] S.D. Desai, S. Giraddi, P. Narayankar, N.R. Pudakalakatti, S. Sulegaon, Back-propagation neural network versus logistic regression in heart disease classification, in: Advanced Computing and Communication Technologies, Springer, 2019, pp. 133–144.
[4] K. Burse, V.P.S. Kirar, A. Burse, R. Burse, Various preprocessing methods for neural network based heart disease prediction, in: Smart Innovations in Communication and Computational Sciences, Springer, 2019, pp. 55–65.
[5] I.K.A. Enriko, M. Suryanegara, D. Gunawan, Heart disease prediction system using k-nearest neighbor algorithm with simplified patient's health parameters, J. Telecommun. Electron. Comput. Eng. (JTEC) 8 (12) (2016) 59–65.
[6] A. Chau, X. Li, W. Yu Liu, Support vector machine classification for large datasets using decision tree and fisher linear discriminant, Future Gener. Comput. Syst. 36 (2014) 57–65, http://dx.doi.org/10.1016/j.future.2013.06.021.
[7] T. Nguyen, A. Khosravi, D. Creighton, S. Nahavandi, Classification of healthcare data using genetic fuzzy logic system and wavelets, Expert Syst. Appl. 42 (4) (2015) 2184–2197.
[8] S. Maji, S. Arora, Decision tree algorithms for prediction of heart disease, in: Information and Communication Technology for Competitive Strategies, Springer, 2019, pp. 447–454.
[9] P.K. Sahoo, S.K. Mohapatra, S.-L. Wu, Sla based healthcare big data analysis and computing in cloud network, J. Parallel Distrib. Comput. 119 (2018) 121–135.
[10] V. Thanigaivasan, S.J. Narayanan, N.C. Sriman Narayana Iyengar, Analysis of parallel svm based classification technique on healthcare using big data management in cloud storage, Recent Pat. Comput. Sci. 11 (3) (2018) 169–178.
[11] Y. Wang, L. Kung, W.Y.C. Wang, C.G. Cegielski, An integrated big data analytics-enabled transformation model: Application to health care, Inf. Manage. 55 (1) (2018) 64–79.
[12] P.M. Kumar, S. Lokesh, R. Varatharajan, G.C. Babu, P. Parthasarathy, Cloud and iot based disease prediction and diagnosis system for healthcare using fuzzy neural classifier, Future Gener. Comput. Syst. 86 (2018) 527–534.
[13] L.R. Nair, S.D. Shetty, S.D. Shetty, Applying spark based machine learning model on streaming big data for health status prediction, Comput. Electr. Eng. 65 (2018) 393–399.
[14] J.K. Harris, N.L. Mueller, D. Snider, D. Haire-Joshu, Peer reviewed: Local health department use of twitter to disseminate diabetes information, united states, Prev. Chronic Dis. 10 (2013).
[15] D.W. Bates, D.J. Cullen, N. Laird, L.A. Petersen, S.D. Small, D. Servi, G. Laffel, B.J. Sweitzer, B.F. Shea, R. Hallisey, et al., Incidence of adverse drug events and potential adverse drug events: implications for prevention, JAMA 274 (1) (1995) 29–34.
[16] R. Thackeray, S.H. Burton, C. Giraud-Carrier, S. Rollins, C.R. Draper, Using twitter for breast cancer prevention: an analysis of breast cancer awareness month, BMC Cancer 13 (1) (2013) 508.
[17] M.J. Paul, M. Dredze, You are what you tweet: Analyzing twitter for public health, in: Fifth International AAAI Conference on Weblogs and Social Media, 2011.
[18] C. Lee, I. Paik, Stock market analysis from twitter and news based on streaming big data infrastructure, in: 2017 IEEE 8th International Conference on Awareness Science and Technology (ICAST), IEEE, 2017, pp. 312–317.
[19] T. Sakaki, M. Okazaki, Y. Matsuo, Earthquake shakes twitter users: real-time event detection by social sensors, in: Proceedings of the 19th International Conference on World Wide Web, ACM, 2010, pp. 851–860.
[20] A. Spark, Apache spark, https://spark.apache.org/, 2019.
[21] A. Kafka, Apache kafka, https://spark.apache.org/, 2019.
[22] S. Nazari, M. Fallah, H. Kazemipoor, A. Salehipour, A fuzzy inference-fuzzy analytic hierarchy process-based clinical decision support system for diagnosis of heart diseases, Expert Syst. Appl. 95 (2018) 261–271.
[23] G. Manogaran, R. Varatharajan, M. Priyan, Hybrid recommendation system for heart disease diagnosis based on multiple kernel learning with adaptive neuro-fuzzy inference system, Multimedia Tools Appl. 77 (4) (2018) 4379–4399.
[24] C.B. Gokulnath, S. Shantharajah, An optimized feature selection based on genetic approach and support vector machine for heart disease, Cluster Comput. (2018) 1–11.
[25] V. Jayaraman, H.P. Sultana, Artificial gravitational cuckoo search algorithm along with particle bee optimized associative memory neural network for feature selection in heart disease classification, J. Ambient Intell. Hum. Comput. (2019) 1–10.
[26] M. Solaimani, M. Iftekhar, L. Khan, B. Thuraisingham, J.B. Ingram, Spark-based anomaly detection over multi-source vmware performance data in real-time, in: 2014 IEEE Symposium on Computational Intelligence in Cyber Security (CICS), IEEE, 2014, pp. 1–8.
[27] Cleveland, S. Hungary, The VA Long Beach, 2019. Heart disease data set. https://archive.ics.uci.edu/ml/datasets/heart+Disease.
[28] T. App, Twitter streaming api, https://developer.twitter.com/en/docs/tweets/filter-realtime/guides/connecting.html, 2019.
[29] scikit, Univariate feature selection, https://scikit-learn.org/stable/modules/feature_selection.html, 2019.
[30] scikit, Chi-squared statistic test, https://scikitlearn.org/stable/modules/generated/sklearn.feature_selection.chi2.html, 2019.
[31] A.U. Haq, J.P. Li, M.H. Memon, S. Nazir, R. Sun, A hybrid intelligent system framework for the prediction of heart disease using machine learning algorithms, Mob. Inf. Syst. 2018 (2018).
[32] V. Wan, W. Campbell, Support vector machines for speaker verification and identification, Vol. 2, 2000, pp. 775 – 784, http://dx.doi.org/10.1109/NNSP.2000.890157.
[33] J. Han, J. Pei, M. Kamber, Data Mining: Concepts and Techniques, Elsevier, 2011.
[34] Z. Bei, Z. Yu, N. Luo, C. Jiang, C. Xu, S. Feng, Configuring in-memory cluster computing using random forest, Future Gener. Comput. Syst. 79 (2018) 1–15.
[35] M. Pal, Random forest classifier for remote sensing classification, Int. J. Remote Sens. 26 (1) (2005) 217–222.

Hager Ahmed obtained a Master's degree in Computer Science in 2017. I obtained a Bachelor's degree in Information systems from the Faculty of Computer and Information, University Assuit, Egypt. I am a researcher member of the Big Data Team in Egypt. My research interests are centered on Big data Analytics, Data Mining, Sentiment Analysis, Natural Language Processing, Machine Learning, and Streaming Data.

Eman Younis is currently working as an Associate Professor at Minia University, Faculty of Computers and Information, Information Systems Department. She got her B.Sc. degree from Zagazig University, Egypt, 2002. She obtained her MSc degree from Meunofia University, Egypt in 2007. She received her Ph.D. degree from Cardiff University, UK in 2014. She spent some time as post-doc at Nottingham Trent University, UK. Her research interests are machine learning, data mining, Geo-spatial data processing, semantic web, sentiment analysis and emotion recognition.

Please cite this article as: H. Ahmed, E.M.G. Younis, A. Hendawi et al., Heart disease identification from patients' social posts, machine learning solution on Spark, Future Generation Computer Systems (2019), https://doi.org/10.1016/j.future.2019.09.056.

Abdeltawab Hendawi obtained his M.Sc. and Ph.D. in Computer Science and Engineering from the University of Minnesota, Twin Cities, and M.Sc. and B.Sc. from Cairo University. His research interest is in the broad area of Data Science with more focus on smart cities related applications. His work has been recognized by a number of awards at the ACM SIGSPATIAL, IEEE MDM, and Blue Sky Ideas. Dr. Hendawi is the founding co-chair of the IEEE Big Spatial Data workshop 2016 to 2019.

Abdelmgeid A. Ali is a Professor in Computer Science Department, Minia University, El Minia , Egypt. He has published over 80 research papers in prestigious international journals, and conference proceedings. He has supervised over 60 Ph.D. and M.Sc. Students. Prof Ali is a member of the International Journal of Information Theories and Applications (ITA). Prof Ali interests are Information Retrieval, Software Engineering, Image Processing, Data security, metaheuristics, IOT, Digital Image Steganography, Data Warehousing.

Please cite this article as: H. Ahmed, E.M.G. Younis, A. Hendawi et al., Heart disease identification from patients' social posts, machine learning solution on Spark, Future Generation Computer Systems (2019), https://doi.org/10.1016/j.future.2019.09.056.