

Prediction Model for Talent Management

Abdul Wadud Alom

Roll No: MCA173019

Department of Computer Science & Engineering



Submitted to: Dr. Saiyed Umer

Department of C.S.E

Aliah University, New town, Action area 2, 700156

DECLARATION

I hereby declare that the project work entitled “**Prediction model for Talent Management**” submitted to the Aliah University, is a record of an original work done by me under supervision of **Dr.Saiyed Umer**, Assistant professor of Computer Science & engineering, Aliah university, and this project work is submitted in the partial fulfillment of the requirements for the award of the degree of Master of Computer Applications. The results embodied in this project report have not been submitted to any other University or Institute for the award of any degree or diploma.

Abdul Wadud Alom

Roll No: MCA173019



ALIAH UNIVERSITY

(UGC, AICTE and NCTE Approved Autonomous Institution)

CERTIFICATE

This is to certify that the project report entitled “**Prediction model for Talent Management**” submitted to **Aliah University** in partial fulfillment of the requirement for the award of the degree of **MASTER OF COMPUTER APPLICATION (MCA)**, is a bonafide and original work carried out by **Mr. ABDUL WADUD ALOM Roll No-MCA173019** under my supervision during the academic year 2020 in Aliah University.

Internal Project Guide

.....

Signature

(Dr.Saiyed Umer)

Assistant Professor,

Dept. of C.S.E

Aliah University, Newtown, Action area 2

University Campus: II-A/27, Action Area II, Newtown, Kolkata, West Bengal, 700156

Website:www.aliah.ac.in,E-mail:info@aliah.ac.in,Ph-No:03323416444

ACKNOWLEDGEMENT

In the accomplishment of this project successfully, many people have best owned upon me their blessings & heart pledged support, this time I am utilizing to thank all the people who have been concerned with this project. Primarily I would thank god for being able to complete this project with success. Then I would thank to my project instructor **Dr.Saiyed Umer**, whose valuable guidance has been the ones that helped me patch this project & make it full proof success. His suggestions & instructions have served as the major contributor towards the completion of the project. Then I would like to thank my parents & friends who have helped me with their valuable suggestions & guidance has been very helpful in various phases of the completion of the project. Last but not the least I would like to thank my classmates who have helped me a lot.

Abstract

Traditionally, Industry practitioner surveys continually report that talent management is very important for recognize talent, it is a big challenge for CEOs organizations. Other prospective to its importance by declaring that talent management can increase organizational performance also refers to the identification, development, appraisal, deployment retention of talented employees. In order to carry out talent management, information technologies can solve this problem more efficiently more accurately. So, In this project we take a data set named “Job classification” from kaggle & we build a prediction model using ‘Python’ where we apply different types of machine learning algorithm to test on which algorithm it gives best result. At last we find the result. So, this model reduces the time of Human Resource Manager to find the result with better proficiency.

Introduction:

e-HRM is electronics human resource management. HRIS is the intermediate phase in the developmental path of HRM i.e. e-HRM. HRIS phase was a phase when the employees were still considered as a resource. The human resource management system and HRIS are a system of storing and sharing information about the "Human Resource" in any organization. The processes through which certain individuals come to be identified as talent can be categorized into three approaches: intuitive, individualized, or systematic. The first approach involves the identification of talented employees via processes that are unstructured and informal with the determination based on the intuitive or "gut-feel" opinion of the executives undertaking the evaluation. The intuitive approach is criticized extensively. High house (2008) argued that the notion of intuitive experience—whereby HR professionals and other key stakeholders can predict human behavior and an employee's likelihood of success—is a myth, while Dries (2013) noted that processes founded on conjectural assumptions or conducted without formal assessment policies may favor individuals similar to the assessor. Intuitive processes therefore are deemed of little strategic value to organizations because identifying talent based on "...in stink and intuition [is] not only inadequate. Traditionally, HRM was performed on papers and was done manually without much use of technology. But with the increasing popularity and usage of software and intranet in other functional areas in the organizations, HR department also felt the need to take advantage of different software based system which in the beginning used very simple data storing technology to the more complex and advanced forms of software for example, ERP system which integrated all the functional departments of an organization. The information needed in HRIS could be accessed by the concerned persons with the use of the intranet within the organization. Then with further technology developments, particularly in the field of internet and web-based technology, the way of doing HRM further changed and today in its current form it is called e-HRM. The e-HRM is a web-based technology that integrates all the HR activities transforming the role of HR managers making them strategic partners in the organizations. For less time consuming & better efficiency we build a prediction model to carry out this job using different types of machine learning algorithm.

To carry out this job we import different types of machine learning classification model such as Logistic Regression, Support Vector Machine, Random Forest, Nearest Neighbor, and Decision Tree. In the first step we collect data from 'kaggle' dataset community then Data preprocessing step comes that is very important for any prediction related work. In this proposed model, the collected dataset is to be preprocessed. The first phase of this preprocessing stage involves data cleaning. In this phase, data cleaning is done to fill up the missing data and to discard any noisy data. Then define the whole data set into two parts i.e. train & test, in this case we divided dataset into 80:20 basis, 80% for training and 20 % for test. After then train the different machine learning classification model, if data fit smoothly then test the model with test data and we get the expected result and if unfortunately data is not fit smoothly then perform the **'train different classification model'** phase again. In experimental section we have seen on Orgimpact attribute SVM gives higher accuracy than other attribute or others model, so we select a candidate on the basis of Orgimpact score so that it will be very effective on recruitment process & with this prediction model we can find out result easily & it is less time consuming & also it reduces the headache of an Human Resource Manager.

Proposed Methodology:

In the first step we collect data from 'kaggle' dataset community then Data preprocessing step comes that is very important for any prediction related work. In this proposed model, the collected dataset is to be preprocessed. The first phase of this preprocessing stage involves data cleaning. In this phase, data cleaning is done to fill up the missing data and to discard any noisy data. Then define the whole data set into two parts i.e train & test, in this case we divided dataset into 80:20 basis, 80% for training and 20 % for test. After then train the different machine learning classification model, if data fit smoothly then test the model with test data and we get the expected result and if unfortunately data is not fit smoothly then perform the **'train different classification model'** phase again.

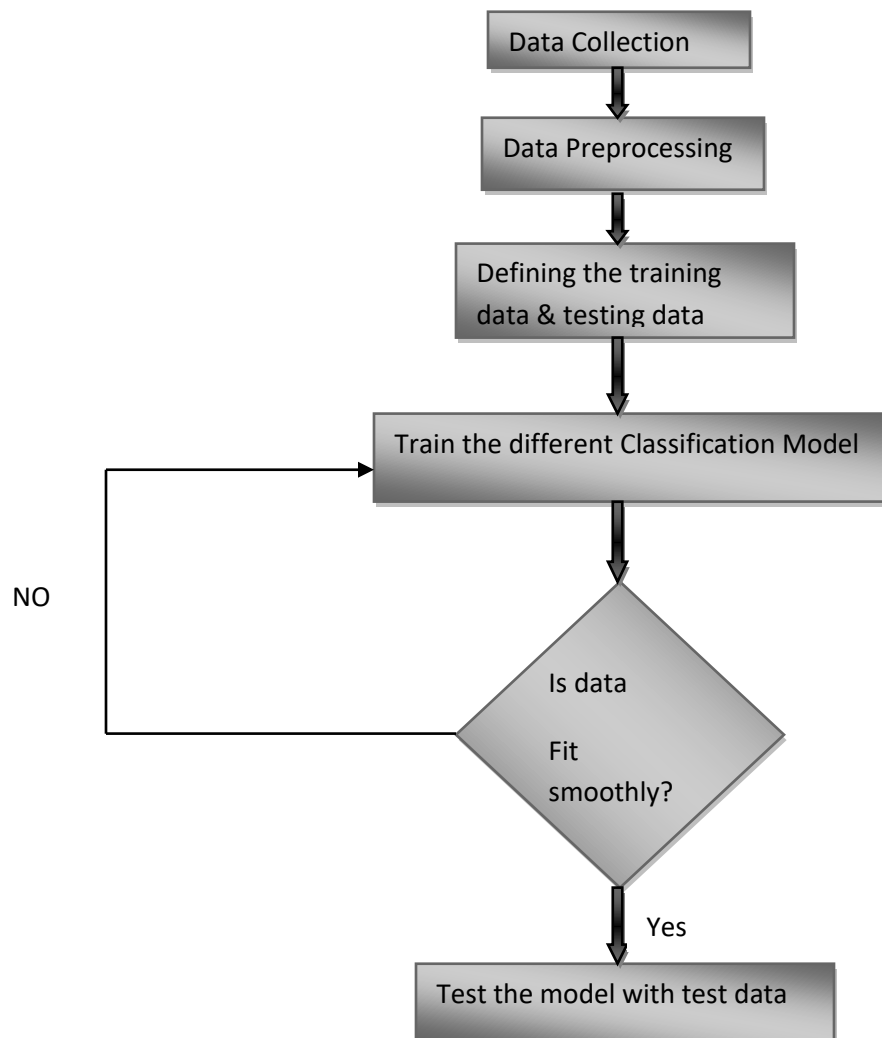


Fig1: Workflow structure of Prediction Model for Talent Management

Dataset:

Job Classification Dataset:

typical job class specs feature information and pay grade .Typically job class specs have information which characterize the job class- its features, and a label- in this case a pay grade - something to predict that the features are related to.




DataSet Characteristics	Multivariate	No. of Instances	845
Attribute Characteristics	Real	No. of Attributes	13
Associated Tasks	Classification	Missing Values?	No



Attributes:

ID,JobFamily,JobFamilyDescription,Jobclass,JobClassDescription,payGrade,Education allLevel,Experience,OrgImpact,ProblemSolving.

Snapshot of Visual representation of attribute:

Job Classification data

ID	# JobFamily	JobFamilyDescri...	# JobClass
		<div>Communications A... 15%</div> <div>Buildings And Facil... 12%</div> <div>Other (48) 73%</div>	
1	1	Accounting And Finance	1
2	1	Accounting And Finance	2
3	1	Accounting And Finance	3
4	1	Accounting And Finance	4
5	2	Administrative Support	5
6	2	Administrative Support	6

Job Classification data			
A JobClassDescrip... 	# PayGrade 	# EducationLevel 	# Experience 
<div>66</div> <div>unique values</div>			
Accountant I	5	3	1
Accountant II	6	4	1
Accountant III	8	4	2
Accountant IV	10	5	5
Admin Support I	1	1	0
Admin Support II	2	1	1
Admin Support III	3	1	2
Administrative Support IV	4	4	0



Data preprocessing:

Data preprocessing transform the raw data into useful and efficient format. In this proposed model collected data set is to be preprocessed. The first phase of this preprocessing stage involves handling categorical data with the help of Label Encoder. Here, the string formatted data are replaced by integer values. Second stage of data preprocessing involves ignoring unnecessary attribute from dataset at the normalization phase. Here, rescaling real valued numeric attribute into the range of $[0, 1]$, to get a better prediction on proposed model.

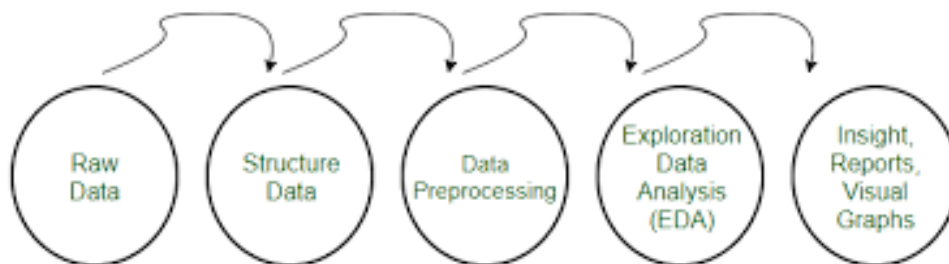


Fig2: Visual Representation of Data preprocessing

Data normalization:

Normalization refers to rescaling real valued numeric attributes into the range [0,1]. It is useful to scale the input attributes for a model to predict better result on proposed model.

$$\text{new value } X' = \frac{\text{original value } x - \min(x)}{\max(x) - \min(x)}$$

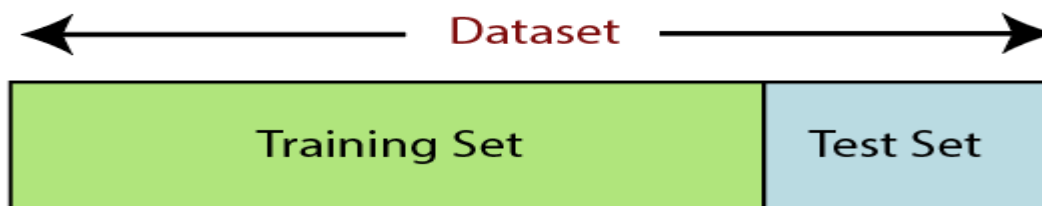
Here are code used for normalization:

```
from sklearn.preprocessing import MinMaxScaler
```

```
X = preprocessing.scale(X)
```

Defining the training data & testing data:

Every dataset for Machine Learning model must be split into two separate sets – training set and test set. Here 80% data are for training data & 20% for Testing.



Here are codes used for define train & test data:

```
from sklearn.model_selection import train_test_split
```

```
X_train,X_test,y_train,y_test=train_test_split(X,y,test_size=0.2,random_state=0)
```

Data classification:

Classification is a process of categorizing a given set of data into classes .It can be performed on both structured and unstructured data. The process starts with predicting the class of given data points. The classes are often referred to as target, label or categories. The classification predictive modeling is the task of approximating the mapping function from input variables to discrete output variables.

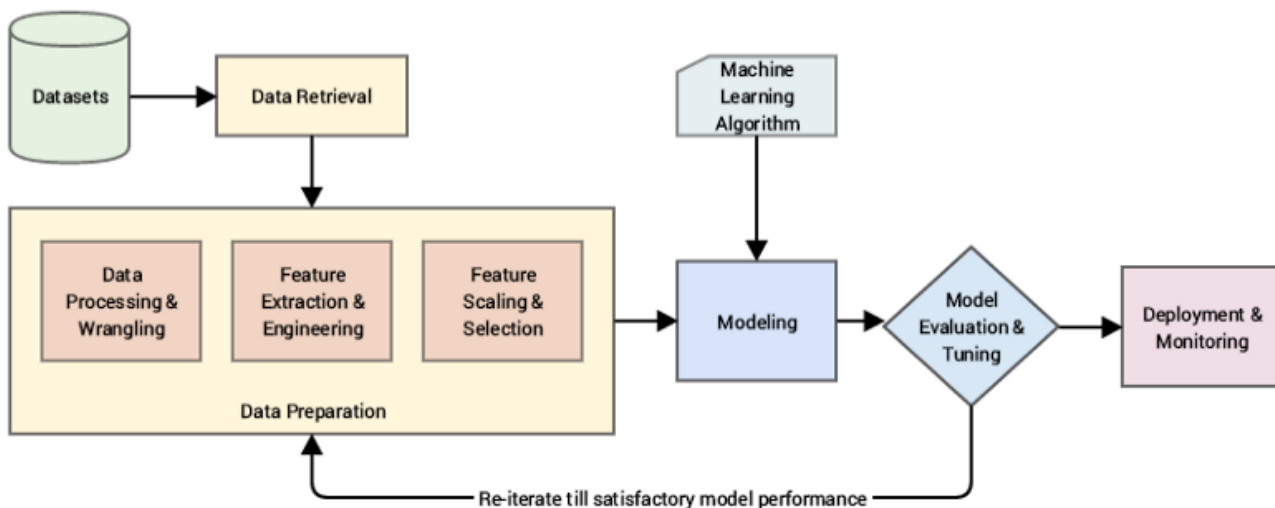


Fig4: A standard machine learning pipeline

Train the different Classification Model:

In this case we apply different types of classification model on train data set these are-

1. Logistic Regression
2. KNN
3. Decision Tree
4. SVM
5. Random Forest

Logistic Regression:

Logistic regression uses an equation as the representation, very much like linear regression. Input values (x) are combined linearly using weights or coefficient values (referred to as the Greek capital letter Beta) to predict an output value (y). A key difference from linear regression is that the output value being modeled is a binary value (0 or 1) rather than a numeric value.

Below is an example logistic regression equation:

$$y = e^{(b_0 + b_1 * x)} / (1 + e^{(b_0 + b_1 * x)})$$

Here are codes for fit logistic regression model on dataset-

```
#logistic regression
```

```
logi=LogisticRegression()
```

```
logi.fit(X_train,y_train)
```

```
predictions=logi.predict(X_test)
```

```
print(accuracy_score(y_test, predictions))
```

```
print(confusion_matrix(y_test, predictions))
```

KNN:

As we know K-nearest neighbors (KNN) algorithm can be used for both classification as well as regression. The following are the recipes in Python to use KNN as classifier as well as regressor.

Here are codes for fit logistic regression model on dataset-

```
#KNeighbor classifier
```

```
classifier_knn=KNeighborsClassifier(n_neighbors=5,metric='minkowski',p=2)
```

```
classifier_knn.fit(X_train,y_train)
```

```
y_pred=classifier_knn.predict(X_test)
```

```
print(accuracy_score(y_test, y_pred))
```

```
print(confusion_matrix(y_test, y_pred))
```

Decision Tree:

It is a non-parametric supervised learning method used for [classification](#) and [regression](#). The goal is to create a model that predicts the value of a target variable by learning simple decision rules inferred from the data features.

Here are codes for fit logistic regression model on dataset-

```
#decision tree
```

```
from sklearn.tree import DecisionTreeClassifier
```

```
cart = DecisionTreeClassifier()
```

```
cart.fit(X_train, y_train)
```

```
predictions = cart.predict(X_test)
```

```
print(accuracy_score(y_test, predictions))
```

```
print(confusion_matrix(y_test, predictions))
```


Support Vector Machine:

It is a set of supervised learning methods use for [classification](#), [regression](#) and [outlier's detection](#).

Here are codes for fit logistic regression model on dataset-

```
sv = SVC(kernel='linear')  
  
sv.fit(X_train, y_train)  
  
predictions = sv.predict(X_test)  
  
print(accuracy_score(y_test, predictions))  
  
print(confusion_matrix(y_test, predictions))
```

Random Forest:

A random forest is a Meta estimator that fits a number of decision tree classifiers on various sub-samples of the dataset and uses averaging to improve the predictive accuracy and control over-fitting

Here are codes for fit logistic regression model on dataset

```
#random forest  
  
rf = RandomForestClassifier(n_estimators=10)  
  
rf.fit(X_train, y_train)  
  
predictions = rf.predict(X_test)  
  
print(accuracy_score(y_test, predictions))  
  
print(confusion_matrix(y_test, predictions))
```

Experimental Coding:

Here are snapshot of sample code:

```
# -*- coding: utf-8 -*-
```

```
"""
```

Created on Thu Feb 20 16:56:04 2020

@author: Abdul

```
"""
```

```
import numpy as np
```

```
import matplotlib.pyplot as plt
```

```
import pandas as pd
```

```
from sklearn.preprocessing import LabelEncoder
```

```
from sklearn.preprocessing import StandardScaler
```

```
from sklearn.metrics import accuracy_score
```

```
from sklearn.metrics import confusion_matrix
```

```
from sklearn.svm import SVC
```

```
from sklearn import preprocessing
```

```
from sklearn.preprocessing import MinMaxScaler
```

```
from sklearn.ensemble import RandomForestClassifier
```

```
from sklearn.neighbors import KNeighborsClassifier
```

```
from sklearn.linear_model import LinearRegression
from sklearn.linear_model import LogisticRegression
```

```
dataset=pd.read_csv('jobclassinfo2.csv')
print(dataset)
```

```
dataset.drop(["JobFamily","JobFamilyDescription","JobClass","JobClassDescription","PayGrade"],axis=1,inplace=True)
```

```
X= dataset.iloc[:, [0,1,3,4,5,6,7]].values
```

```
y=dataset.iloc[:,2].values
```

```
X = preprocessing.scale(X)
```

```
from sklearn.model_selection import train_test_split
```

```
X_train,X_test,y_train,y_test=train_test_split(X,y,test_size=0.2,random_state=0)
```

```
#logistic regreesion
```

```
logi=LogisticRegression()
```

```
logi.fit(X_train,y_train)
```

```
predictions=logi.predict(X_test)
```

```
print(accuracy_score(y_test, predictions))  
print(confusion_matrix(y_test, predictions))
```

```
#KNeighbor classifier
```

```
classifier_knn=KNeighborsClassifier(n_neighbors=5,metric='minkowski',p=2)  
classifier_knn.fit(X_train,y_train)  
y_pred=classifier_knn.predict(X_test)  
print(accuracy_score(y_test, y_pred))  
print(confusion_matrix(y_test, y_pred))
```

```
#decision tree
```

```
from sklearn.tree import DecisionTreeClassifier  
cart = DecisionTreeClassifier()  
cart.fit(X_train, y_train)  
predictions = cart.predict(X_test)  
print(accuracy_score(y_test, predictions))  
print(confusion_matrix(y_test, predictions))
```

```
#svm

sv = SVC(kernel='linear')

sv.fit(X_train, y_train)

predictions = sv.predict(X_test)

print(accuracy_score(y_test, predictions))

print(confusion_matrix(y_test, predictions))

#random forest

rf = RandomForestClassifier(n_estimators=10)

rf.fit(X_train, y_train)

predictions = rf.predict(X_test)

print(accuracy_score(y_test, predictions))

print(confusion_matrix(y_test, predictions))
```

Experimental Results:

Classifier	Without Normalization (%)	With Normalization (%)
Logistic Regression	0.7142	0.2857
KNN	0.4285	0.2857
Decision Tree	0.7143	0.7142
SVM	0.5	0.4285
Random Forest	0.42	0.5

Fig5: target level as Pay Grade

Classifier	Without Normalization (%)	With Normalization (%)
Logistic Regression	0.4285	0.5
KNN	0.3571	0.4285
Decision Tree	0.64	0.7142
SVM	0.3571	0.3571
Random Forest	0.7142	0.5714

Fig6: target level as Education Level

Classifier	Without Normalization (%)	With Normalization (%)
Logistic Regression	0.6428	0.6428
KNN	0.5714	0.7857
Decision Tree	0.5714	0.6428
SVM	0.8571	0.8571
Random Forest	0.7142	0.7857

Fig7: target level as Organization Impact

Classifier	Without Normalization (%)	With Normalization (%)
Logistic Regression	0.5	0.5714
KNN	0.6428	0.6428
Decision Tree	0.7857	0.7142
SVM	0.7142	0.7142
Random Forest	0.7142	0.7857

Fig8: target level as Problem Solving.

Discussion:

At first we perform preprocessing into data set & then set independent & dependent variable for further process. Then we split data set into train & test, there is 80% for training & 20% for test, then import different types of classifier model such as Logistic Regression, Support Vector Machine etc from scikit learn & pass the data through these models & find out accuracy in percentage form. In experimental section we have seen on Orgimpact attribute SVM gives higher accuracy than other attribute or others model, so we select a candidate on the basis of Orgimpact score so that it will be very effective on recruitment process & with this prediction model we can find out result easily & it is less time consuming & also it reduces the headache of an Human Resource Manager.

Conclusion:

We go through various observations on dataset & lots of experimental test. At the end we have seen on Orgimpact attribute SVM gives higher accuracy (85.71%) than other attribute or others model, so we select a candidate on the basis of Orgimpact score so that it will be very effective on recruitment process & with this prediction model we can find out result easily & it is less time consuming & also it reduces the headache of an Human Resource Manager. In the future, if similar studies are conducted to generate the dataset used in this report, more feature vectors need to be calculated so that the classifiers can form a better idea of the problem at hand.

References:

1. <https://www.kaggle.com/HRAlyticRepository/job-classification-dataset>
2. <https://scikit-learn.org/stable/>
3. "Elements of Statistical Learning: Data Mining, Inference, and Prediction. 2nd Edition. Datasets: Coronary Heart Disease Dataset." Elements of Statistical Learning: Data Mining, Inference, and Prediction. 2nd Edition. Accessed April 27, 2016. <http://statweb.stanford.edu/~tibs/ElemStatLearn/>.
4. https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LogisticRegression.html
5. Dean, Jeffrey, and Sanjay Ghemawat. "MapReduce: simplified data processing on large clusters." *Communications of the ACM* 51.1 (2008): 107-113.
6. <https://digitalguardian.com/blog/what-data-classification-data-classification-definition#:~:text=Data%20classification%20is%20broadly%20defined,easier%20to%20locate%20and%20retrieve.>
7. <https://scikit-learn.org/stable/modules/svm.html>
8. <https://builtin.com/data-science/random-forest-algorithm>
9. <https://www.edureka.co/blog/machine-learning-algorithms/>
10. <https://acknowledgementsample.com/2020/02/12/sample-acknowledgement-for-a-final-year-project/>
11. <https://www.analyticsvidhya.com/blog/2015/09/build-predictive-model-10-minutes-python/>
12. <https://towardsdatascience.com/top-10-algorithms-for-machine-learning-beginners-149374935f3c>
13. <https://towardsdatascience.com/understanding-random-forest-58381e0602d2>
14. <https://medium.com/@Synced/how-random-forest-algorithm-works-in-machine-learning-3c0fe15b6674>
15. https://www.tutorialspoint.com/machine_learning_with_python/machine_learning_with_python_classification_algorithms_random_forest.htm
16. <https://machinelearningmastery.com/supervised-and-unsupervised-machine-learning-algorithms/>
17. <https://www.geeksforgeeks.org/ml-types-learning-supervised-learning/>
18. <https://www.geeksforgeeks.org/data-preprocessing-machine-learning-python/>
19. <https://www.analyticsvidhya.com/blog/2020/04/feature-scaling-machine-learning-normalization-standardization/>
20. <https://machinelearningmastery.com/rescaling-data-for-machine-learning-in-python-with-scikit-learn/>