Edge Hill University

THE DEPARTMENT OF COMPUTER SCIENCE CIS3156 INTELLIGENT SYSTEMS

Level 6

COURSEWORK 2

2019/2020

Jakub (Jacob) Strykowski 24325457

KEYWORDS

- Machine learning
- Sentimental analyses
- Natural language processing
- Decision tree
- KNN
- SVM
- Random forest
- Bag-of-word

CONTENTS

INTRODUCTION

Refer to (Sajid, 2019) every day in a world is generated 2.5 quintillion bytes of data, the interpretation collected information is too much for human cognitive abilities. Newest machines have computing power many times overgrow humans possibilities, however without human conducting computers are useless. Data analyze is sometimes named 'data mining', because it is similar to mining, miners most of time dig huge amount worthless rock to find little diamond. The diamonds in opinion analyze is extracted from comments costumer's opinion about product or service, the data miners also demand pickaxe – artificial intelligent classificator.

Companies save feedback about their products and analyze it, because they would like to try to make conclusions how to improvement of their services or product. The size of data collected in this reasons is yearly growing, in that reason to analyze information is need the tool to predict costumers feeling based about their feedback in comments. The one from possibly solutions of this task is to use machine learning algorithms. Machine learning algorithms are seen like one of the most perspective branch of intelligent technology (IT) sector. Machine learning have two main branches: supervised learning using classification based on abstraction or previous learned knowledge and unsupervised learning based on clustering class data on categories to help process of decision making.

Data can be group by features which are depending from target of experiment, it can be for example gender effect on bought products or how age effects on probability of an car accident.

One of main problems to solve in data analyzes is pattern recognitions, where patterns can be observe using mathematical algorithms, especially machine learning technics. The algorithms classify collected data based on statistics technics an example is pattern recognition used in devices with speech recognition, speaker identification or medical diagnosis (sakilAnsari, nd). The pattern is necessary to make generalize and makes abstracts model for example the pronunciations of worlds is different, but know systems like Apple Siri or Google speech-to-text can recognize the words from human speech and answer to those words or even transform speech to text.

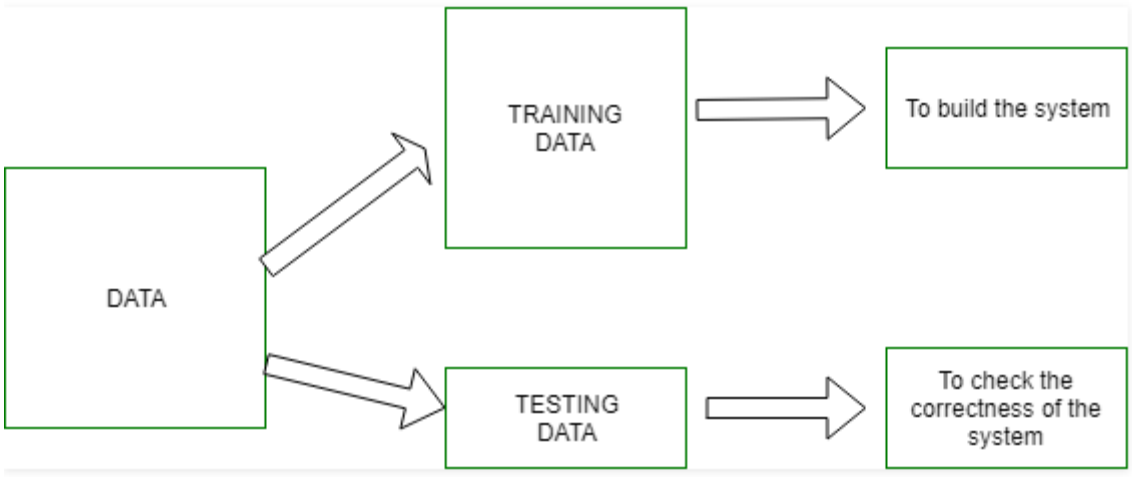


Figure 1

Process of predicting the categories of given data instances is called classification, it based on machine learning algorithm called 'classificator' which would be used in experiment part of this work. Commonly used example is spam and not spam classification, it contains two categories (classes): 'spam' and 'non-spam' (Asiri, 2018). Generally the classificator uses training data to learn connection between features for example sex, blood pressure, presence or absence of illness symptom would be used to classification to diagnosis patient, the answer might be healthy or have an illness (Wikipedia, 2019).
Figure 1 shows up how typical machine learning algorithm works, firstly data set is split into training data and test data, the reason of test data is to check how effective and precisely is the examined algorithm, because the target of machine learning is to make autonomous decision using algorithm learnt from training data. The ability to make prediction based on learnt knowledge can be seemed to be one of biggest advantage of machine learning, in addition solutions like Recursive Neural Network can self-learn and improve the their characteristics with time.

The artificial intelligent classificator to make prediction needs learn which features shows up customers emotions, there is also necessity of pattern recognition, because people demonstrate their feeling in lots of different ways and the generalization is crucial. A solution of this problem could both complex or simple, the opinion could be score for example 5 and 4 positive, 2 neutral, 1 and 0 negative, it is more accurate, however more costly than just polarity classifying in two class positive or negative (Asiri, 2018). An everyday example of different methodology in collecting user's feedback is the different between Facebook and Instagram video chat quality scoring system. Facebook asks users to vote from 1 to 5 and if user chosen 1 (the lowest), the system would ask the user about issue, Instagram shows up only two options: positive or negative and in case of negative do not always ask about background. That different in systems might demand on budgets of those companies, Facebook is much richer than Instagram and can spend on customer surveys more resources. The more advance scoring system allows Facebook to easier conduction updates, changes or crisis management, because their rapidly and exactly know which part of video chat do not work. On the other hand that accurate costumer opinion analyses is quite expensive and Instagram scoring system has this advantage that it is cheaper and more clear for user, because it has only positive and negative options. Instagram maybe would not know as precisely as Facebook about feedback or would not notice slowly decreasing quality service, but they also could fast react to crisis like system crash. Next important part of sentimental analyze is to answer the questions: what features the opinion has? Opinion could has a lot of features, the most important one are: subject, opinion holder and answer it is subjective or objective. The machine learning field of extracting information like that is called Natural Language Processing, LNP is task requiring knowledge from linguistics, computer science, information engineering and artificial intelligence (wikipedia, 2020). Machine learning algorithms are the best known tool for LNP, ML algorithms has fundamental advantage causing from their process of learning they focus on most common cases and easier than human can find patterns or create new rules. The popular technique used in LNP is word embedding, it demands on mapping to each word a vector which contain information about meaning of this word, but presents with numbers in N dimensional spaces. Sentimental analyze, natural language processing, machine learning are 'hot' topic in industry, the biggest company like Google, Apple develops applications,

systems and build word banks to make result as good as it is possible. Intelligent systems could be seen like one of the most prospective field to plumb for computer science student.

METHODOLOGY

This task presents analysis of opinion or review mining an important field of artificial intelligence. In further part of this paper would be present process of development and evaluation machine learning algorithms used to sentimental analysis opinions about video games. The data are based on Amazon web-store where customers could publish their opinion about bought products. The dataset contains 10000 customers opinions and it is split into two sets: training set and test set. The training set would be used to learn an algorithm and test set would be used to evaluate the algorithm.
Two main factors used to test effectiveness are precision and recall. Recall answers a question "HOW COMPLETE THE RESULTS ARE" and precision 'HOW USEFUL THE SEARCH RESULT ARE'. Both precision and recall are important, however often increasing one of them decrease second, in most cases is needed balance between those parameters, in that reason was invented F1-Score. F1-score is counting with mathematic formula and it present relationship between algorithm's precision and recall, in that reason it is better than accuracy which also could be used for evaluation. The feature used to comparation between each algorithm is weighted average value of F1-Score.

Process of choosing two best algorithms for this piece of work started from analysis a Stanford paper (Richard Socher, Alex Perelygin, Jean Y. Wu, Jason Chuang,, nd), it describes advance machine learning technic - Recursive Neural Network, it seems to be one of the best solution for sentimental analyses, because learning process is repeated as long as result are the best. The ability to understand context is easy for human, but an artificial intelligence has problem with it and basic algorithm do not provided context analyses. RNN can analyze contexts in sentences, it is an major advantage and makes RNN good tool to natural language processing. The Stanford's research describe their RNN and for comparation they choose decision tree based on Stanford's words bank (Stanford, nd) and SVM. The complexity of write RNN exceed requirements of this task, but it seems like good destination for further self-developing. Stanford scientist used decision tree and SVM in one row with RNN, it produce the idea of applying those algorithms in this task. In theory results should be good, because large decision tree covers large percent of possible cases. The label has only two possible cases: positive and negative, this simplification would good effects on linear variant of SVM, because in this task SVM can effortlessly splits the N-dimensional space(N – numbers of features) (Drakos, 2018). On the other hand, both algorithm would not analyze text's context, because it is not kept in provided dataset, but this disadvantage make coding easier and may not strongly effect on final outcome. Instead of decision tree in experiment would be used algorithm called Random Forest seemed to be more effective, Random Forest used a lot of randomly generated decision tress based on provided features.

The hypothesis is that Random Forest and SVM are the best algorithms from module called Intelligent System and to confirm it the experiment would contain implementations of KNN, decision tree, SVM and Random Forest and combination of both RF and SVM in majority vote. As a comparator would be use value of weighed average F1-Score which presents differences in efficient between algorithms. Also, it could be assume that KNN would present low score, because data combined a lot of different features (in comparation of overall number of instances) presenting unique words, it would affected with mixed data and noise.

Majority vote

Majority vote could being called ‘plurality voting’ is and ensemble method using to improve result of single machine learning algorithm. The algorithm uses voting to calculate the result, the answer would be the prediction which was predicted by majority of voting classificators. In an experiment would be apply majority vote with tree classificators: two Random Forest and one linear SVM. In Figure 20 is presented example of majority voting of four diverse predictors (classificators) took a part in voting, tree voted for class 1 and only one voted for class 2, in that reason result of majority voting equals 1.

Support Vector Machine (SVM)

SVM is commonly use machine language algorithm for classification and regression analysis in supervised learning model. The learning process used vectors to find out the large margin separators. Margin is defined by minimal distance between hyperplane and nearest point representing record from dataset (Figure 21) (Shai Shalev-Shwartz, Shai Ben-David, 2014). SVM is using extreme hyperplane to calculate the optimal hyperplane (Figure 22), also dimension of plane depends on numbers of features and it is less by one, for example for two features it is just a line (Drakos, 2018). Figure 23 shows up how SVM works in case of tree labels presented with different colors for each, SVM linear use linear functions to split dimensional space, it might be not effective when data are mixed. In case when data are group the linear SVM is more cost-effective that non-linear SVM, because in this experiment are only two labels a better choice is linear SVM.

Referring to (Yadav, 2018) the SVM has those advantages:

- Good effectivity in high dimension.
- Effectivity is good in case when number of features is larger than number of training instances.
- SVM could be used in extreme case of binary classification.

Disadvantages:

- It is not time-effective when dataset is large, and the this characteristics gets worse with increasing of datasets.
- Have a problem with overlapped classes.
- More advanced kernel function could be confusing.

As could be seen from previous advantages and disadvantage list the SVM suits in background of this task, the features makes 2000 dimension space (it is high dimension) and the proportion between instances and features is low, the number of features is high.

Random Forest

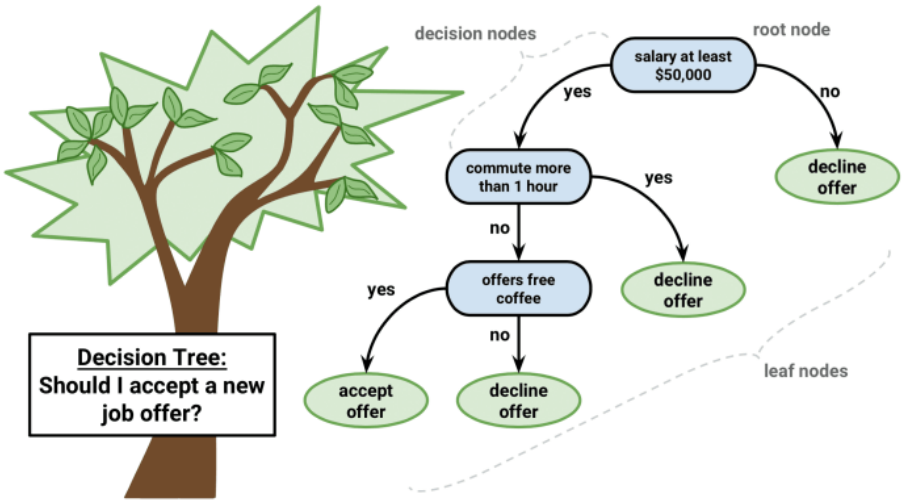


Figure 2

Random forest is one from most famous machine learning algorithms, it is using many random decision trees and make from those trees an 'Random Forest' . Decision tree is an simple machine learning algorithm whish also can be used to presentation the process of decision, as can be on Figure 2, decision tree presents data in way which is easy for human to understand the concept. Decision tree observations about an item are represented in the branches, item's target value is representing in the leaf. A leaf node represents a class label it follows that classification rules is the path from root to leaf (Brid, 2018). Figure 24 shows up simple example of decision tree, the tree present the decision tree algorithm using red and blue numbers and underlining. Root answers for question is red? There are two possibility yes and no, no answer is leaf and return four blue zeros. If answers was yes, the next step would be the branch with question: it is underline? both answers are leaf and would end algorithm with results: it is underline would return two red, underline one numbers, if no it would return red zero (Yiu, 2019). Random Forest is similar to majority vote because it also is an ensemble, it use significant number of decision tree learnt randomly (number of features is also random and it is less than or equals to max_features) chosen features from provided dataset. Figure 25 shows up how this algorithms makes predictions, firstly generates forest, secondly makes voting, finally return the prediction based on voting result. In Figure 25 could be seen that the winner in voting is class equals one because six decision trees predicted this class when just tree decision trees which predicted zero.
The scikit learn implementation of random forest has features (scikit-learn, nd):

- Estimator what is an number of trees in forest, by default this value is assign to 100.
- Max_features number of features consider, by default sqrt(n_features) = 44 for this dataset.

Advantages:

- Compares to advance supervised algorithm decision tree represents competitive performance

- The voting in training process is cross-validation, what could save cost because in sufficient situations errors would be eliminated during voting.

Disadvantages:

- Based on randomization and F1-Score is not constant.
- The Random Forest is less interpretable than an individual decision tree.
- With increasing numbers of trees in forest the computational costs also significant increase.

Provided data

Features in those datasets are represents by words in sentence, the data are save in typical for excel file .csv and contain information about how many each word appeared in each sentence. The Label is 0 or 1, 0 for negative opinion, 1 for positive opinion about video game where opinions are represents in reviews from Amazon web-shops and have predefined libels. Data are present in two different ways one of them is raw data with full sentence and second is numbers about how many times each word appears in the instance (tokenized). Uploading the data based on method presented on lecture to upload tokenized version data. Tokenization is way to present words or sentence, there are two types of tokenization first split text into sentence, second split sentences into tokens. A token is extracted from sentence word for example "The quick brown fox jumps over the lazy dog" ->
['The', 'quick', 'brown', 'fox', 'jumps', 'over', 'the', 'lazy', 'dog'] the ‘dog’ is token (Anon., nd). Bag-of-ward saves information about how many time unique token appeared in sentence. It is an simple representation, but useful in NLP. For example of implementation (wikipedia, 2019) are those two sentences:
(1) John likes to watch movies. Mary likes movies too.
(2) Mary also likes to watch football games.
Tokenized version:
"John","likes","to","watch","movies","Mary","likes","movies","too"
"Mary", "also", "likes", "to", "watch", "football", "games"
And finally bag-of-words:
BoW1 = {"John":1,"likes":2,"to":1,"watch":1,"movies":2,"Mary":1,"too":1};
BoW2 = {"Mary":1,"also":1,"likes":1,"to":1,"watch":1,"football":1,"games":1};
As can be seen each unique token has assigned value of repetition in the sentence. The data provided in csv file are saved with bag-of-word algorithm.

Figure 3 shows up how tokenized data are presented in Excel file and for example compare first instance to raw text sentence.

frustratin	frustratio	fully	fun	function	funny	future	gain	game	gameboy
0	0	0	0	0	0	0	0	2	0
0	0	0	0	0	0	0	0	12	0
0	0	0	0	0	0	0	0	3	0

Figure 3

if you have played this game in the arcade or on a dreamcast unit you know how fast and colorful a fighting game this one is . truth is it is one of the best 2d figthers you can play . here is the bad news though , if you own a playstation and want this translation you will be disapointed when you play . first of all the lack of the tag feature really does kill this game . without being able to tag in another teamate mid battle it takes away some of the games statagy . i think it also seems to move and load much slower on the playstation than in the dreamcast . we just have to face it , this game is a next generation game on a five year old piece of hardware , and it just does not live up to its true potential . i love my plastation , but i have to advise all to pass on marvel vs . capcom for it . play street fighter alpha 3 instead if you insist on getting the best 2d fighting action you can out of your old playstation box .

Figure 4

Figure 4 presents full sentence in example word game appeared two times and value in Figure 3 for this record is also two. The word bank in this dataset has 2000 unique words and 5000 instances in training set and 5000 instances in test set, each instance has label's value 1 if opinion is positive or 0 if opinion is positive. The provided data were previous prepared by tutor and this dataset could be built by human, however the algorithm, after learning from the training set would make their own predictions.

```
data from dataset and use function load_from_cvs to build a classifier.
def load_data_from_csv(input_csv):
     df = pd.read_csv(input_csv, header=0)
     csv_headings = list(df.columns.values)
     feature_names = csv_headings[:len(csv_headings) - 1]
     label_name = csv_headings[len(csv_headings) - 1:len(csv_headings)][0]
     df = df._get_numeric_data()
     numpy_array = df.as_matrix()
     number_of_rows, number_of_columns = numpy_array.shape
     instances = numpy_array[:, 0:number_of_columns - 1]
     labels = []
     for label in numpy_array[:, number_of_columns - 1:number_of_columns].tolist():
         labels.append(label[0])
     return feature_names, instances, labels
```

Figure 5

This method (Figure 5) open csv file with data, read data and return as a result feature_names, instances and labels, next those values are used in further building classifier.

Libraries

The experiment is based on sklearn library, this library provides ready implementation of KNN, decision tree, classification_raport, Random Forests and SVM. For making a classifier was used mlxtend library and for data import was used torch library.

```
In [153]
import pandas as pd
from sklearn.neighbors import KNeighborsClassifier
import numpy as np
from torch.utils.data import DataLoader, Dataset
import torch.optim as optim
from tqdm import tqdm
from sklearn.model_selection import train_test_split
import mlxtend
from sklearn import tree
from sklearn.metrics import classification_report
from sklearn.ensemble import RandomForestClassifier
from sklearn import svm
from mlxtend.classifier import StackingClassifier
import matplotlib.pyplot as plt
from sklearn.ensemble import VotingClassifier
```

Figure 6

This experiment require use python libraries: pands, sklearn, mlxtend, numpy, torch, tqdm (Figure 6).

EXPERIMENTS

The experiment presentation is ordered with increasing F1-score, because validation of algorithms needs reference point.

Reference point

KNN

First loop is for generally searching for efficient peaks, the peak would show up which value of n, number of checking nearest neighbors, is with the best parameters.

```
In [166]: for i in range(1, 40, 5):
              classifier = KNeighborsClassifier(n_neighbors=i)
              classifier.fit(X=training_instances, y=training_labels)
              predicted_test_labels = classifier.predict(test_instances)
              print(i)
              print(classification_report(test_labels, predicted_test_labels, digits=3)[310:315])

          1
          0.585
          6
          0.637
          11
          0.605
          16
          0.639
          21
          0.595
          26
          0.628
          31
          0.602
          36
          0.624
```

Figure 7

From Figure 7 can be assumed that the peak is around the n equals 6, next step is to probe surroundings of this n. The loop step equaled 5 because Knn is greedy algorithm and integration by one would be computation costly and time consuming.

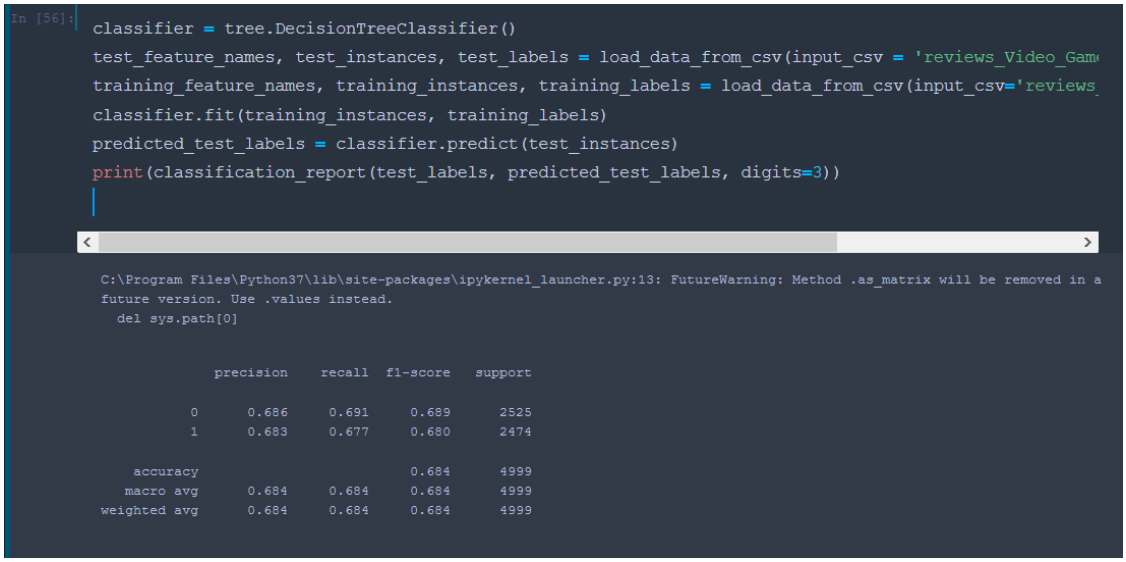
```
In [167]: for i in range(3, 9):
              classifier = KNeighborsClassifier(n_neighbors=i)
              classifier.fit(X=training_instances, y=training_labels)
              predicted_test_labels = classifier.predict(test_instances)
              print(i)
              print(classification_report(test_labels, predicted_test_labels, digits=3)[310:315])

          3
          0.602
          4
          0.624
          5
          0.610
          6
          0.637
          7
          0.602
          8
          0.634
```

Figure 8

Figure 8 shows up that (with precision to integer) best characteristics of KNN is when n equals 6 and then F1-Score equals 0.637.

DECISION TREE

Decision tree implementation and result are present on figure below.



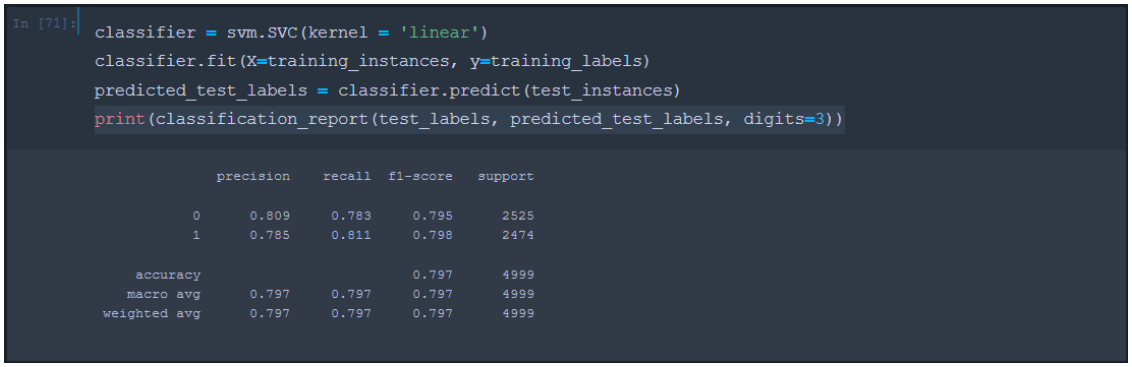
```
In [56]: classifier = tree.DecisionTreeClassifier()
         test_feature_names, test_instances, test_labels = load_data_from_csv(input_csv = 'reviews_Video_Gam
         training_feature_names, training_instances, training_labels = load_data_from_csv(input_csv='reviews_
         classifier.fit(training_instances, training_labels)
         predicted_test_labels = classifier.predict(test_instances)
         print(classification_report(test_labels, predicted_test_labels, digits=3))

C:\Program Files\Python37\lib\site-packages\ipykernel_launcher.py:13: FutureWarning: Method .as_matrix will be removed in a
future version. Use .values instead.
  del sys.path[0]

              precision    recall  f1-score   support

           0      0.686     0.691     0.689      2525
           1      0.683     0.677     0.680      2474

    accuracy                          0.684      4999
   macro avg      0.684     0.684     0.684      4999
weighted avg      0.684     0.684     0.684      4999
```

Figure 9

Figure 9 shows up result of decision tree and the result of weighted avg of F1-score equals 0.684. The decision tree had better result than KNN with the beast value of n_neighbors.

SVM

First from testing algorithms is SVM, an advance machine learning solution using N-dimension space to interpret provided data and used hyperplanes to classification o regression.



```
In [71]: classifier = svm.SVC(kernel = 'linear')
         classifier.fit(X=training_instances, y=training_labels)
         predicted_test_labels = classifier.predict(test_instances)
         print(classification_report(test_labels, predicted_test_labels, digits=3))
```

	precision	recall	f1-score	support
0	0.809	0.783	0.795	2525
1	0.785	0.811	0.798	2474
accuracy			0.797	4999
macro avg	0.797	0.797	0.797	4999
weighted avg	0.797	0.797	0.797	4999

Figure 10

Figure 10 shows up effectiveness of SVM, f1-score equals 0.797 and is higher than KNN's or decisions trees' score.

Random Forest

First step provide general research of algorithm and f1-sfcore monotonicity of F1_score function, this knowledge would be use to set the best value for max_feature(Figure 11). An disadvantage on Random Forest is huge cost for large forest, if iteration was by one, it would be very computer power demining and not effective. The solution for this problem is make one loop with step and after that make look with step one with smaller range surrounding values where F1-Score can be the best.

```
In [141]: res = []
          for i in range(1, 70, 5):
              classifier = RandomForestClassifier(n_estimators=100, max_features = i, n_jobs = 1)
              classifier.fit(training_instances, training_labels)
              predicted_test_labels = classifier.predict(test_instances)
              res.append(float(classification_report(test_labels, predicted_test_labels, digits=3)[310:315]))
              print(i)
              print(classification_report(test_labels, predicted_test_labels, digits=3)[310:315])

          #float(classification_report(test_labels, predicted_test_labels, digits=3)[310:315])

          1
          0.782
          6
          0.813
          11
          0.820
          16
          0.816
          21
          0.812
          26
          0.808
          31
          0.809
          36
          0.807
          41
          0.811
          46
          0.803
          51
          0.802
          56
          0.805
          61
          0.801
          66
          0.798
```

A

Figure 11

```
In [142]: xs = []
          for i in range(0, len(res)):
              xs.append(i)

          plt.scatter(xs, res)
          plt.show()
```

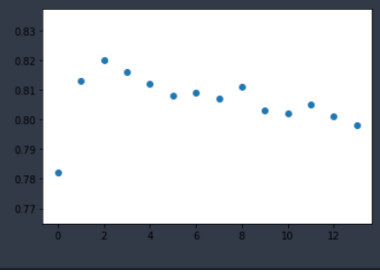


Figure 12

Figure 12 presents graph of F1score function for Random Forest algorithm, it could be seen two peaks one in 11 and second in 41. First peak is represents the highest value 0.820 and second is around the auto mode value for random forest classifier (scikit-learn, nd).

```
In [144]: res = []
          for i in range(6, 16):
              classifier = RandomForestClassifier(n_estimators=100, max_features = i, n_jobs = 1)
              classifier.fit(training_instances, training_labels)
              predicted_test_labels = classifier.predict(test_instances)
              res.append(float(classification_report(test_labels, predicted_test_labels, digits=3)[310:315]))
              print(i)
              print(classification_report(test_labels, predicted_test_labels, digits=3)[310:315])

          #float(classification_report(test_labels, predicted_test_labels, digits=3)[310:315])

          6
          0.805
          7
          0.810
          8
          0.810
          9
          0.813
          10
          0.811
          11
          0.815
          12
          0.814
          13
          0.809
          14
          0.808
          15
          0.810
```

Figure 13

Figure 13 is an confirmation of previous research of peak and results with information that the best value for max_features is 11 and then F1-Score equals 0.815.
The different in score value between Figure 13 and Figure 12 is consequence of randomization of learning process and it needs more observation.

```
In [148]: xs = []
          for i in range(0, len(value11)):
              xs.append(i)

          plt.scatter(xs, value11)
          plt.show()
```

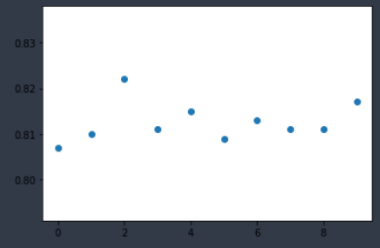


Figure 14

Figure 14 shows up how F1-Score change without changing the factors, it is slightly different which is after-effect of usage random features to make prediction.

It is interesting that when max features equals the number of instances result was better than just decision tree which is using all features during learning process (Figure 15).

```
In [64]: classifier = RandomForestClassifier(n_estimators=100, max_features = 2000, n_jobs = 1)
         classifier.fit(training_instances, training_labels)
         predicted_test_labels = classifier.predict(test_instances)
         print(classification_report(test_labels, predicted_test_labels, digits=3))
```

```
                precision    recall  f1-score   support

             0      0.748     0.800     0.773      2525
             1      0.780     0.725     0.751      2474

      accuracy                          0.763      4999
     macro avg      0.764     0.762     0.762      4999
  weighted avg      0.764     0.763     0.762      4999
```

Figure 15

Majority vote – two Random Forests and SVM

The desire to constant improvement motivated to one more trail, the result of majority vote with two Random Forest and SVM is the best overall and equals 0.830. An important fact is that max_features are set on peak values, which characterize with highest factors of measurements. Two RF and SVM was chosen because those algorithm present the best results in previous part.

```
In [170]: subclassifier1 = svm.SVC(kernel = 'linear')
          subclassifier2 = RandomForestClassifier(n_estimators=100, max_features = 11, n_jobs = 1)
          subclassifier3 = RandomForestClassifier(n_estimators=100, max_features = 41, n_jobs = 1)
          estimators = [('subc1', subclassifier1),('subc2', subclassifier2),('subc3', subclassifier3)]
          voting_classifier = VotingClassifier(estimators=estimators,voting='hard')
          voting_classifier.fit(training_instances, training_labels)
          predicted_test_labels = voting_classifier.predict(test_instances)
          print(classification_report(test_labels, predicted_test_labels, digits=3))
```

```
                precision    recall  f1-score   support

             0      0.823     0.846     0.834      2525
             1      0.838     0.814     0.826      2474

      accuracy                          0.830      4999
     macro avg      0.830     0.830     0.830      4999
  weighted avg      0.830     0.830     0.830      4999
```

Figure 16

The importance of data segmentation

```
In [71]: classifier = svm.SVC(kernel = 'linear')
         classifier.fit(X=training_instances, y=training_labels)
         predicted_test_labels = classifier.predict(test_instances)
         print(classification_report(test_labels, predicted_test_labels, digits=3))

                      precision    recall  f1-score   support

                   0      0.809     0.783     0.795      2525
                   1      0.785     0.811     0.798      2474

            accuracy                          0.797      4999
           macro avg      0.797     0.797     0.797      4999
        weighted avg      0.797     0.797     0.797      4999
```

Figure 17

Swapped training and test set, the score is a little better what is based for the assumption about overlapping of words in sets, in test set are more sentences with words which makes SVM predict better. It shows up how important is data analyze and preparing of training or test datasets. For simple example there might be two sentences, first one give information how to predicts second, and both are in test set and that lack of knowledge (no sentece in training set) would negatively affect learning process.

```
In [150]: classifier = svm.SVC(kernel = 'linear')
          classifier.fit(X=training_instances, y=training_labels)
          predicted_test_labels = classifier.predict(test_instances)
          print(classification_report(test_labels, predicted_test_labels, digits=3))

                       precision    recall  f1-score   support

                    0      0.810     0.782     0.796      2475
                    1      0.793     0.820     0.807      2524

             accuracy                          0.801      4999
            macro avg      0.802     0.801     0.801      4999
         weighted avg      0.802     0.801     0.801      4999
```

Figure 18

RESULTS with elements of conclusions

At the end in Table1 one and in Figure 19 are presented results of ran machine learning algorithms. It could be seen that different between the worst KNN and the best Majority Vote is significant. The conclusion from this experiment is importance of choice an appropriate machine learning algorithm to solve the task. Sentimental analyze needed tokenization of sentences and to get satisfying result are require more advances algorithms like SVM and Random Forest. SVM is good solution when repetition of result is bigger priority than slightly better F1-Score, because Random Forest based on randomization and the results are not consistent. The Majority Vote with Random Forest (max_features = 11), Random Forest (max_features = 41) and linear SVM generated the best result which was equal 0.830 . The cause of that rewarding outcome is voting (cross-validation) which decreasing errors, because prediction of majority vote is true even one from tree used algorithms generated a wrong prediction.

Algorithm	F1-Score
KNN	0.637
Decision tree	0.684
SVM	0.797
Random Forest (max_features = 41)	0.811
Random Forest (max_features = 11)	0.820
Majority Vote two RF and SVM	0.830

Table 1

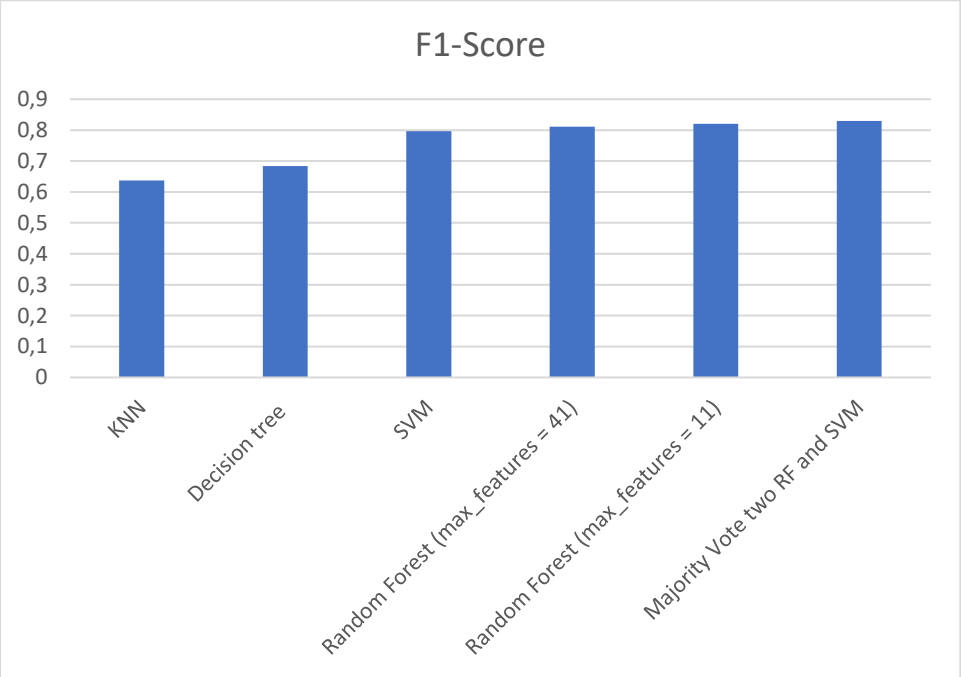


Figure 19

Figure 19 shows up the sticking out result of KNN, this algorithm has problem with task with mixed data, in that reason generate lowest F1-Score from all tested tools. The data in this task have a lot of feature in compare to number of instances, what's make data diverse.

BRIEFLY CONCLUSION

The paper is an answer to problem of sentimental analyses, the introduction describe used algorithms and machine learning names, the methodology explains choice of algorithms with explanation how they works and the experiment shows up how the coding part was conducted.

Results of the experiment are satisfying because F1-Score equaling to 0.830 is very good score. Also both random forest and SVM gratifying scores, on the other hand they have high computational cost. If computational power were crucial factor, the decision tree would be the best algorithm.

REFERENCES

Anon., n.d. [Online].

Anon., nd. *Tokenization.* [Online]
Available at: http://mlwiki.org/index.php/Tokenization
[Accessed 3 January 2020].

Asiri, S., 2018. *Machine Learning Classifiers.* [Online]
Available at: https://towardsdatascience.com/machine-learning-classifiers-a5cc4e1b0623
[Accessed 02 01 2020].

Drakos, G., 2018. *Support Vector Machine vs Logistic Regression.* [Online]
Available at: https://medium.com/@george.drakos62/support-vector-machine-vs-logistic-regression-94cc2975433f?
[Accessed 2 12 2020].

Richard Socher, Alex Perelygin, Jean Y. Wu, Jason Chuang,, nd. *Recursive Deep Models for Semantic Compositionality.* [Online]
Available at: https://nlp.stanford.edu/~socherr/EMNLP2013_RNTN.pdf
[Accessed 02 1 2020].

Sajid, H., 2019. *Deep Learning for Sentiment Analysis.* [Online]
Available at: https://towardsdatascience.com/deep-learning-for-sentiment-analysis-7da8006bf6c1
[Accessed 2 01 2020].

sakilAnsari, nd. *Pattern Recognition | Introduction.* [Online]
Available at: https://www.geeksforgeeks.org/pattern-recognition-introduction/
[Accessed 31 12 2019].

scikit-learn, nd. *3.2.4.3.1. sklearn.ensemble.RandomForestClassifier.* [Online]
Available at: https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html
[Accessed 2 1 2020].

Shai Shalev-Shwartz, Shai Ben-David, 2014. *Understanding Machine Learning: From Theory to Algorithms.* First ed. Cambridge: Cambridge Univeristy Press.

Stanford, nd. *Sentiment Analysis.* [Online]
Available at: https://nlp.stanford.edu/sentiment/
[Accessed 02 1 2020].

wikipedia, 2019. *Bag-of-words model.* [Online]
Available at: https://en.wikipedia.org/wiki/Bag-of-words_model
[Accessed 3 January 2020].

Wikipedia, 2019. *Statistical classification.* [Online]
Available at: https://en.wikipedia.org/wiki/Statistical_classification
[Accessed 1 02 2029].

wikipedia, 2020. *Natural language processing.* [Online]
Available at: https://en.wikipedia.org/wiki/Natural_language_processing
[Accessed 03 01 2020].

Yadav, A., 2018. *SUPPORT VECTOR MACHINES(SVM).* [Online]
Available at: https://towardsdatascience.com/support-vector-machines-svm-c9ef22815589
[Accessed 03 01 2020].

Yiu, T., 2019. *Understanding Random Forest.* [Online]
[Accessed 3 January 2020].

Brid, R. S., 2018. Introduction to Decision Trees. [Online] Available at: https://medium.com/greyatom/decision-trees-a-simple-way-to-visualize-a-decision-dc506a403aeb [Accessed October 2019].
Nicholson, C., 2019. Evaluation Metrics for Machine Learning - Accuracy, Precision, Recall, and F1 Defined. [Online].

APPENDIX

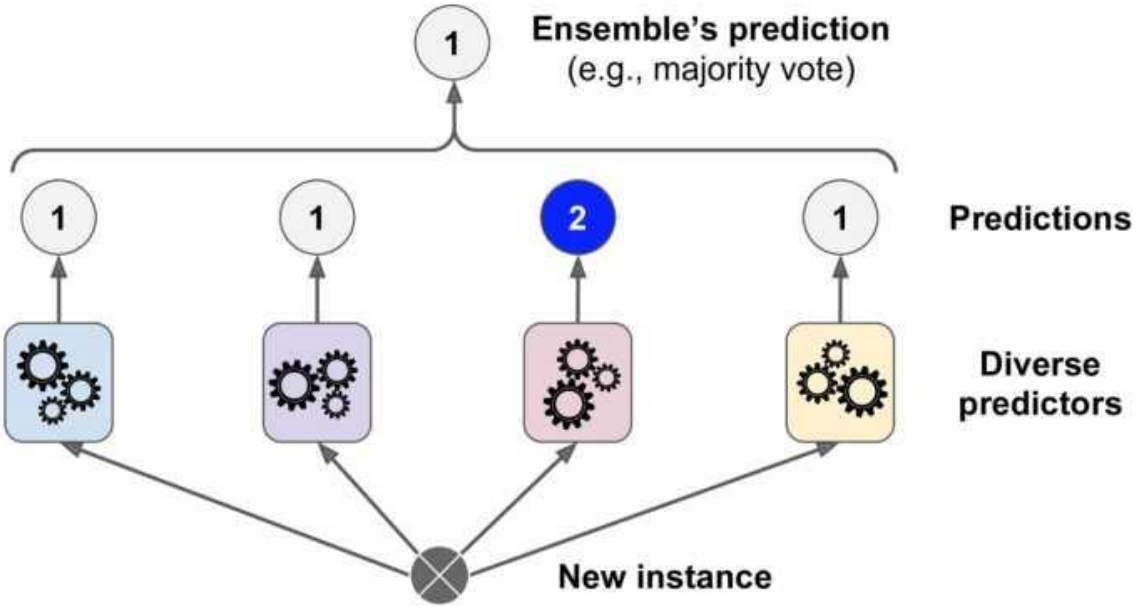


Figure 7-2. Hard voting classifier predictions

知乎 @孙铭泽

Figure 20

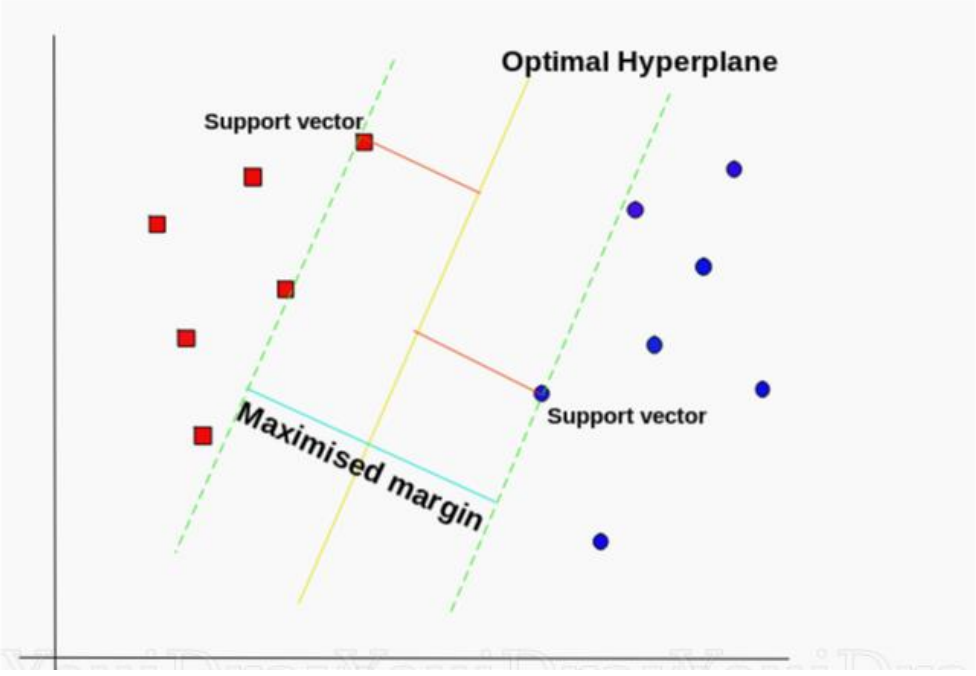


Figure 21

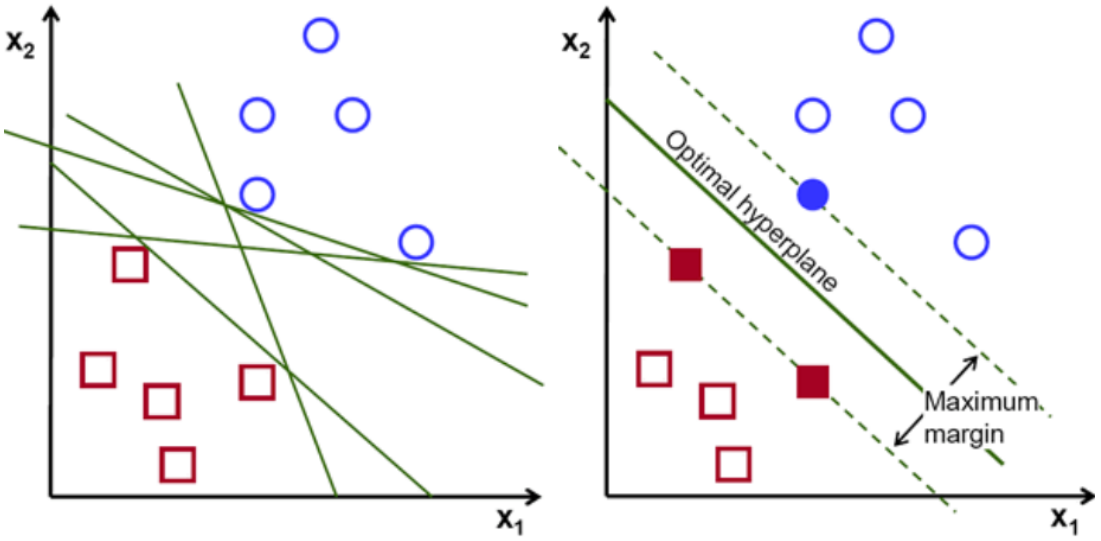


Figure 22

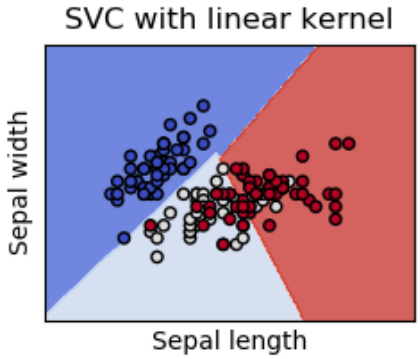


Figure 23

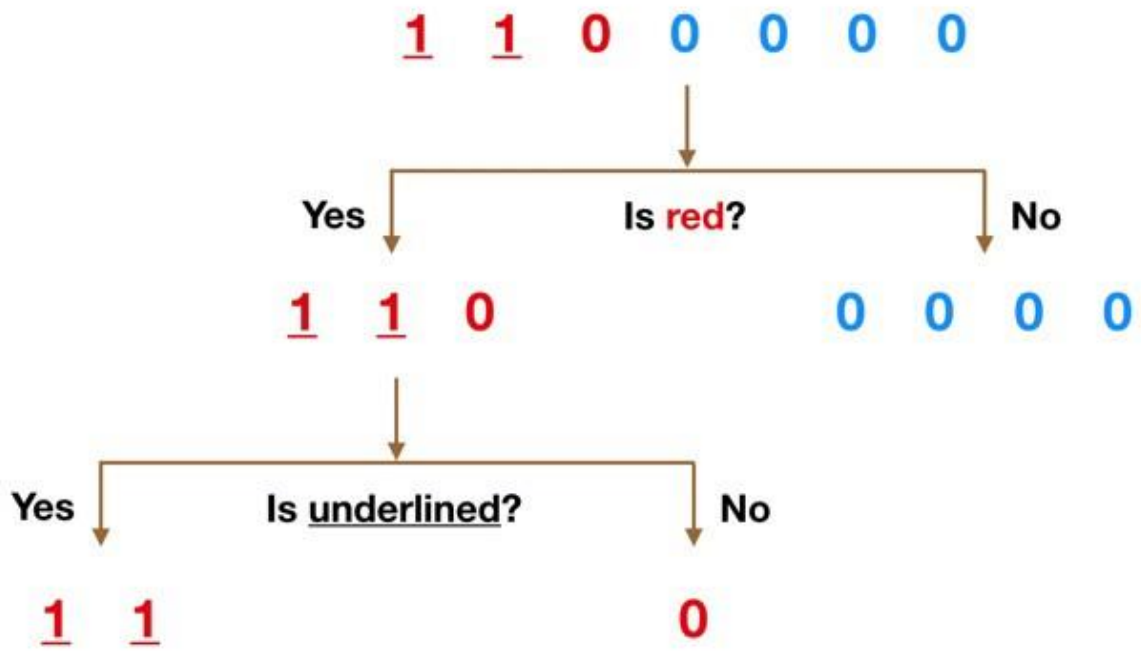


Figure 24 (Yiu, 2019)

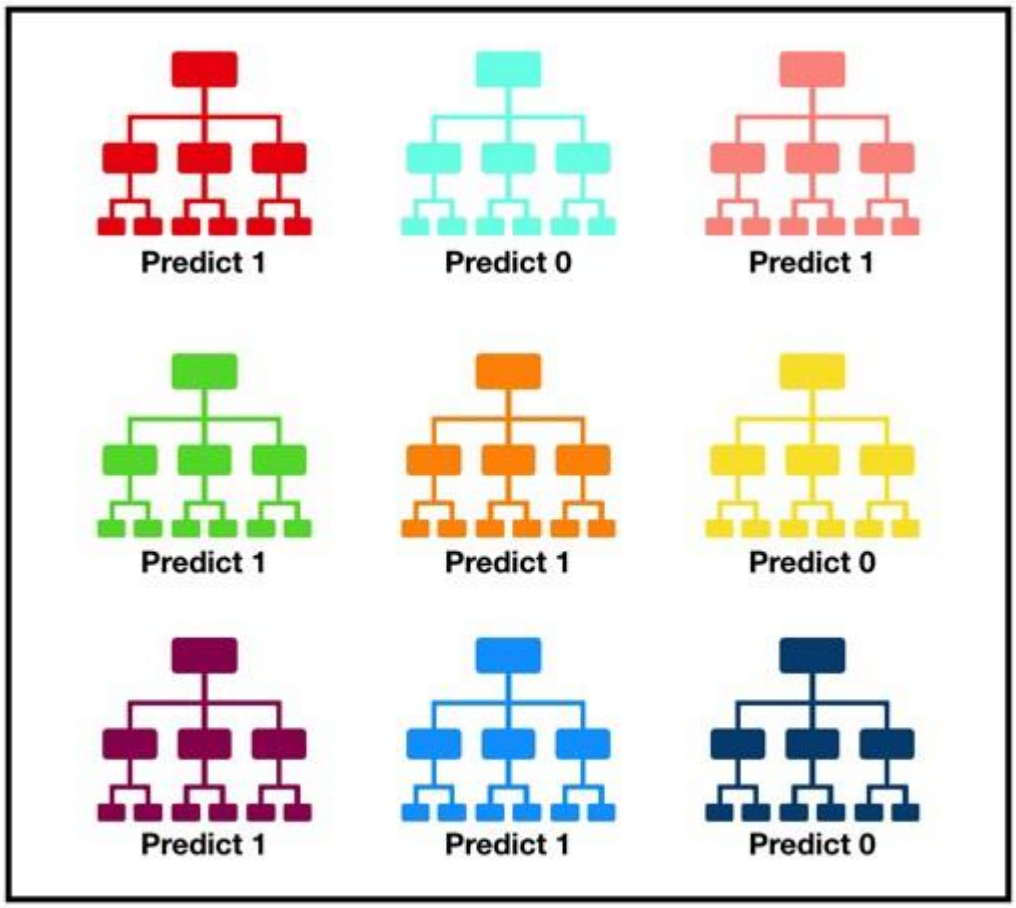

Predict 1
Predict 0
Predict 1
Predict 1
Predict 1
Predict 0
Predict 1
Predict 1
Predict 0
Tally: Six 1s and Three 0s
Prediction: 1

Figure 25