Credit Cards Approval Prediction

Documentation

- Abdelrahman Mahmoud Shaheen
- Mahmoud Bakr
- Mahmoud Nasser

Supervised by:

ENG: Mohamed Hanafy

ITI Graduation Project

Credit Cards Approvals

Brief:

Our data is talking about credit cards approval, many banks spend a lot of time in order to take a decision either approving the credit card for a customer or not.

Some parameters affect this decision like gender, Age, Debt, Married, Bank Customer, Education Level, Ethnicity, Years Employed, Prior Default, Employed, Credit Score, Driver's License, Citizen and Income.

our target is to implement a machine learning code which helps in taking the decision programmatically in order to save time.

Before implementing the machine learning models and choose which model giving the best accuracy, we have to inspect the data, clean the data without losing its consistency then preprocess the data to be used for the machine learning models.

We have independent variables which was numbered from A1 to A15 and the dependent variable is A16 which is approved or not.

Presence of a dependent variable with zero or one forced us to use classification

Models of machine learning.

In order to enhance our project we used grid search for each model to get the best parameters that should be used as arguments for the model to give the best accuracy.

Finally we used Ensembling as a try to get better and better accuracy for the model.

Our code was implemented in the sequence of:

1- Loading data

2- Inspecting data

3- Cleaning data

4- Preprocessing of data

5- Machine learning models:

- Logistic regression
- Decision tree
- Random forest
- K nearest neighbors

6- Hyperparameters tuning (grid search) for each model

7- Ensembling:

- Voting ensemble
- Average ensemble

We will discuss each point

• Loading data

We load the data into a data frame named cc_apps referring to credit cards approvals and changed the names of columns from ("Age") to A1 and so on.

This step was done to show all the columns when printing the data frame and inspecting the data in a good way.

• Inspecting data

1- We found that the column A14 which refers to " Zip Code " is a useless column as it contains high number of codes including zeros which is unapplicable and may damage our model in encoding.

2- We found that we have 690 rows and 15 columns.

3- Some columns we found its type is object instead of float which is the correct type as Age column.

4- Instead of nulls we found this symbol " ? " which need to be replaced with mean or mode according to the variable type.

5- No duplicates was found

• Cleaning data

1- Dropping Zip Code column.

2- Changing the type of columns that have a wrong type like Age column.

3- Replacing the symbol " ? " with mean for the numerical columns and mode with categorical columns.

• Preprocessing of data

1- Splitting the data into 3 data frames, one for the numerical columns, one for categorical columns to deal with it and one for the dependent variable.

2- Encoding the dependent variable using LabelEncoder from sklearn.preprocessing to be zeros and ones and can be used with the machine learning models

3- Encoding the categorical data frame using One Hot Encoder as it contains more than two values.

4- Dropping the main columns from the data frame in order to make all the columns zeros and ones.

5- Concatenating the numerical data frame and the categorical data frame again to be used for the machine learning algorithms into a data frame named "to_ML" referring to machine learning.

6- splitting the data into train 70% and test 30%.

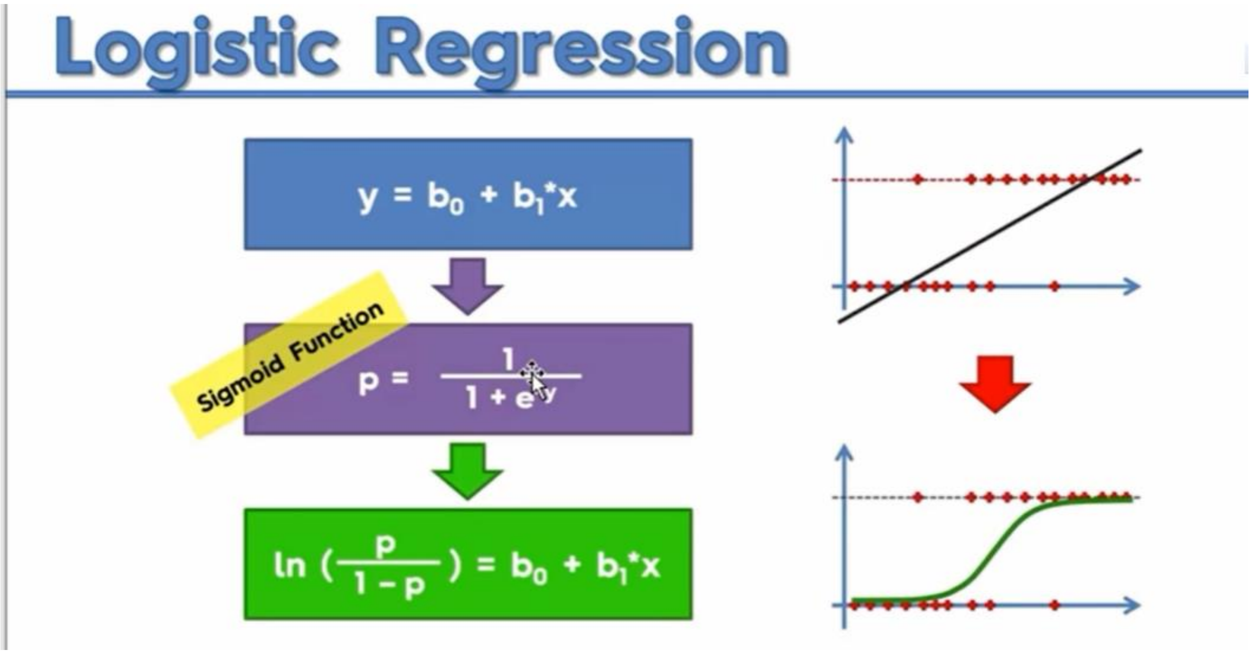
7- so we have x_train , x_test , y_train , y_test.

8- we used standard scaler from sklearn.preprocessing as we should make scaling of the data to make the points of distance between variables applicable.

• Machine learning models

1- Logistic Regression

1- Logistic Regression is the probability of a variable to be zero or one depending on solving two equations to get a new equation which draws "S" curve used to get the probability of each point.



Logistic Regression

$y = b_0 + b_1 {}^* x$

$p = \frac{1}{1 + e^{-y}}$

$\ln\left(\frac{p}{1 - p}\right) = b_0 + b_1 {}^* x$

2- We imported LogisticRegression class from sklearn.linear_model

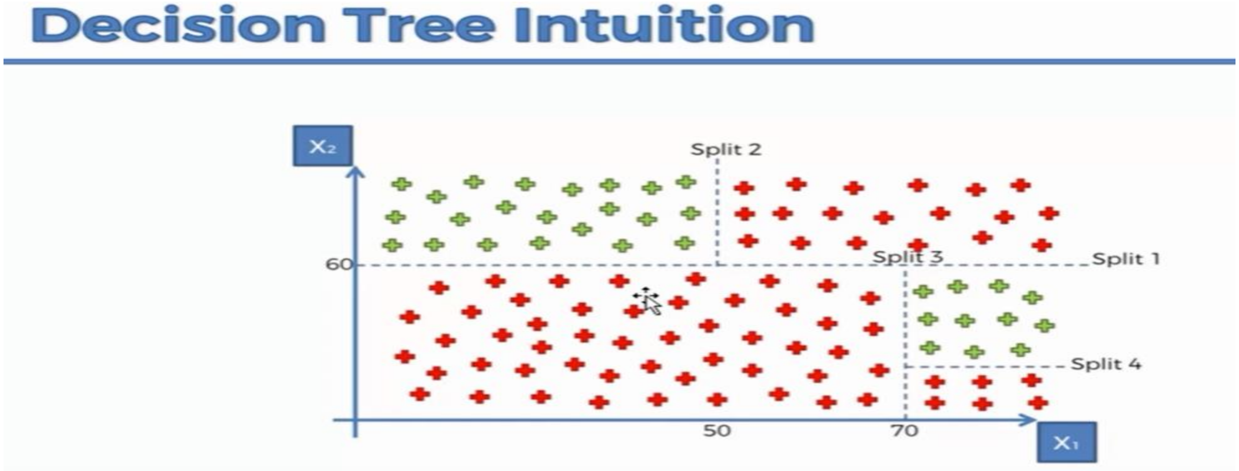
3- The accuracy of logistic regression model was 82.12%

4- In order to improve the accuracy we are going to use grid search (hyperparameters tuning) to get the best parameters which will improve the model's accuracy.

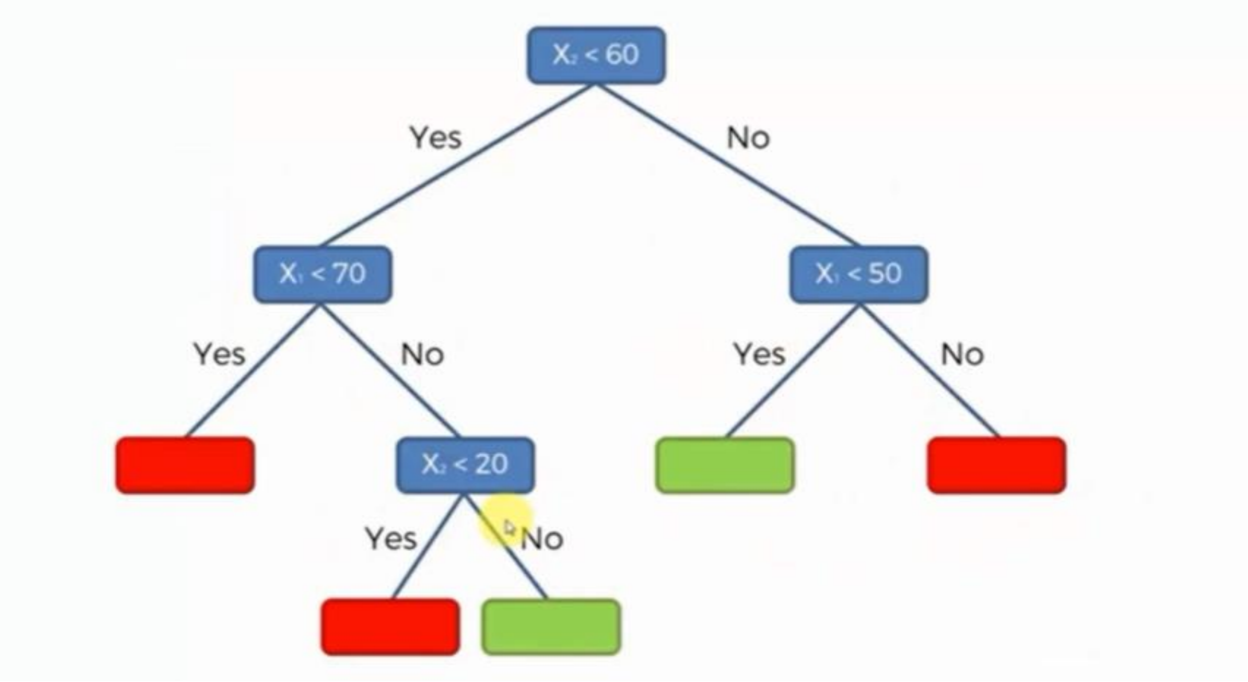
5- The best parameters are (C=0.01, random_state=42, solver='newton-cg')
6- After using these parameters with the values we have gotten we found that the accuracy improved to be 84.54%.

2- Decision Tree

1- The algorithm is based on splitting the data , it tries to put the maximum number of points in one cluster or leaf , splitting the data is related to a physical parameter called entropy which used in thermo dynamics and in brief it refers to random distribution of particles , decision tree was an old school which improved to Random Forest .


Decision Tree Intuition

The tree:



2- We imported DecisionTreeClassifier class from sklearn.tree , the parameters which we used is criterion="entropy" and random_state=42 where criterion parameter is the function used to measure the quality of a split of tree nodes and it is "gini" or "entropy" , entropy is more complex since it makes use of algorithms so the calculation of the Gini Index will be faster.

3- The accuracy of Decision Tree model was 81.15%

4- In order to improve the accuracy we are going to use grid search (hyperparameters tuning) to get the best parameters which will improve the model's accuracy.

5- The best parameters are (max_depth=2, random_state=42 and criterion ="entropy")

6- After using these parameters with the values we have gotten we found that the accuracy improved to be 84%.

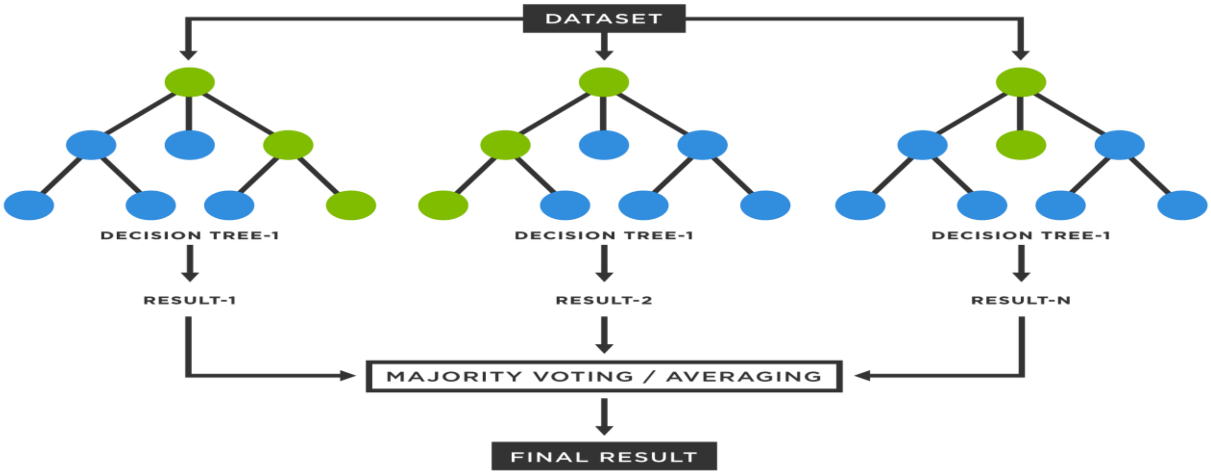
3- Random Forest

1- it aims to use many machine learning algorithms to get a powerful algorithm , Random Forest can be used with different types of machine learning algorithms.

When we used Decision Tree , only one tree was implemented while in Random Forest we have many trees where we take the voting from every tree

Each tree votes for the new data point for which class it belongs or there is another method which is the average of trees

Random forest is used in facial recognition.



2- We imported RandomForestClassifier class from sklearn.ensemble.

3- The accuracy of random forest model was 84%.

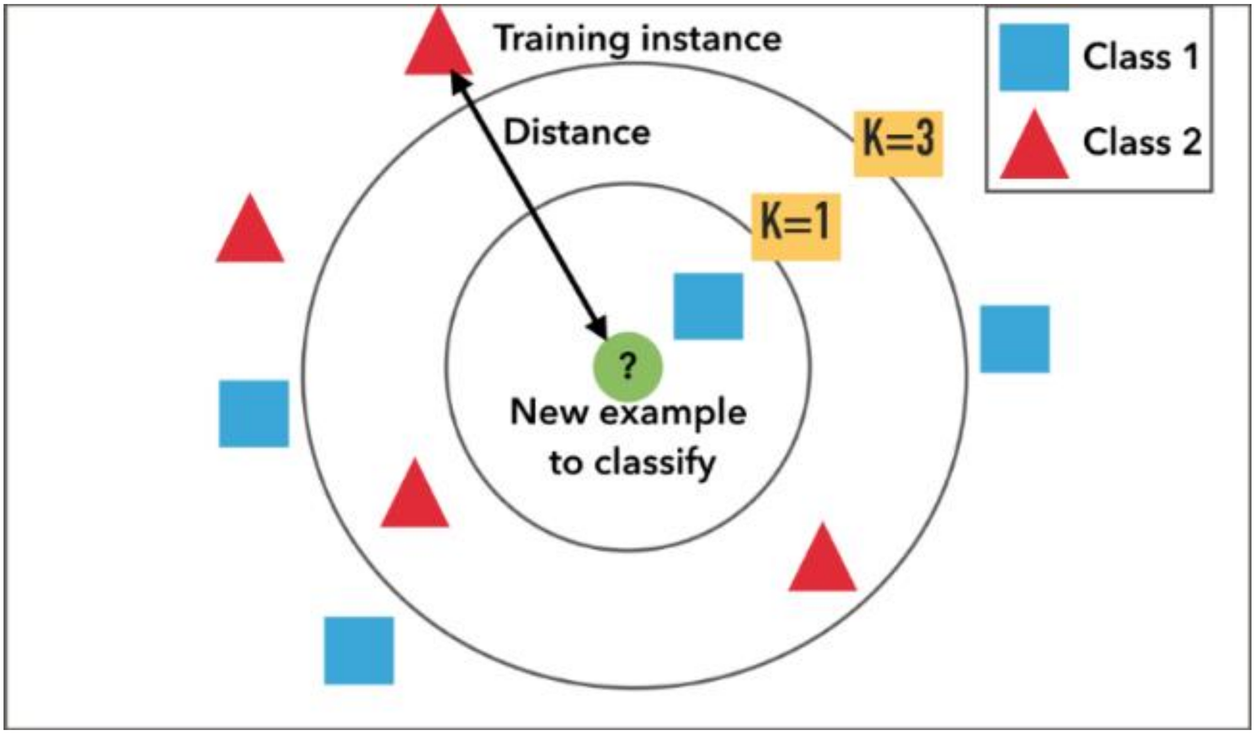
4- In order to improve the accuracy we are going to use grid search (hyperparameters tuning) to get the best parameters which will improve the model's accuracy.

5- The best parameters are (max_depth=8, n_estimators=200, random_state=42)
- n_estimators is the number of trees we want to build before taking the maximum voting or averages of predictions.

6- After using these parameters with the values we have gotten we found that the accuracy improved to be 85.9%.

4- k-nearest neighbors

(KNN) algorithm is a simple, supervised machine learning algorithm that can be used to solve both classification and regression problems. It's easy to implement and understand, assumes the similarity between the new case/data and available cases and put the new case into the category that is most similar to the available categories.



2- We used KNeighborsClassifier from sklearn.neighbors

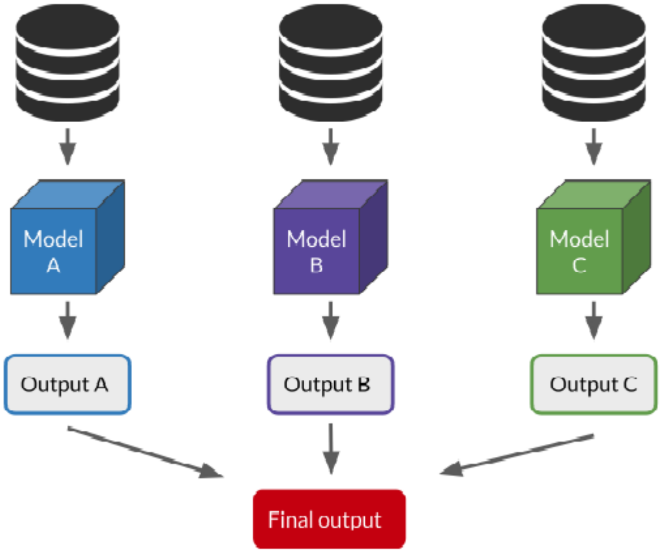
3- The accuracy of KNN model was 82,61 %

4- In order to improve the accuracy we are going to use grid search (hyperparameters tuning) to get the best parameters which will improve the model's accuracy.

5- The best parameters are (n_neighbors=43)
6- After using these parameters with the values we have gotten we found that the accuracy improved to be 85,99 % .

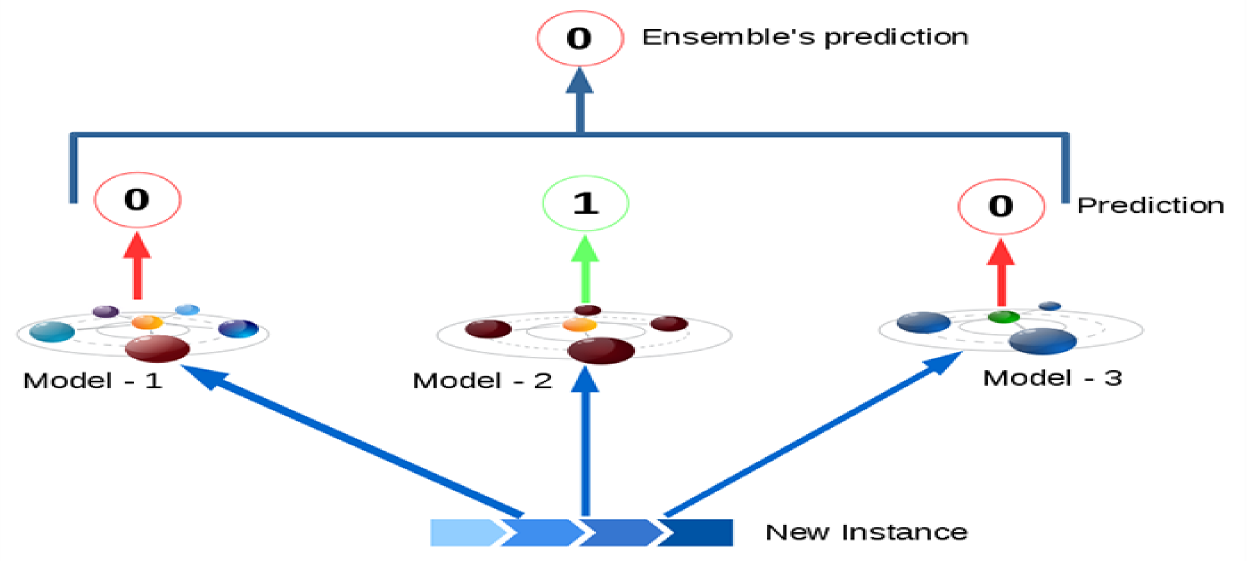
• Ensemble Methods

are techniques that create multiple models and then combine them to produce improved results. Ensemble methods usually produces more accurate solutions than a single model would.



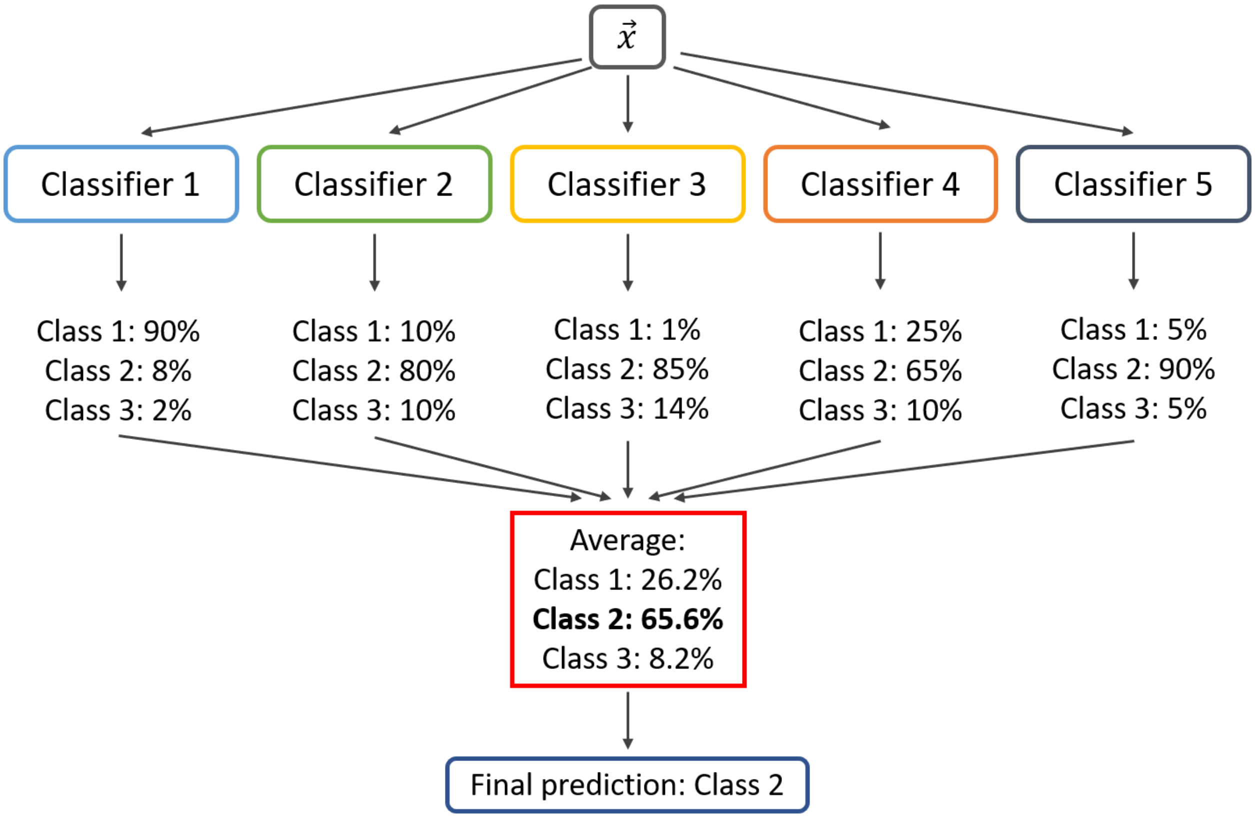
1- Voting ensemble method

the predictions for each label are summed and the label with the majority vote is predicted.



2- Averaging ensemble method

for every instance of test dataset, the average predictions (mean of predicted probabilities) are calculated.



$\vec{x}$
Classifier 1
Classifier 2
Classifier 3
Classifier 4
Classifier 5
Class 1: 90%
Class 2: 8%
Class 3: 2%
Class 1: 10%
Class 2: 80%
Class 3: 10%
Class 1: 1%
Class 2: 85%
Class 3: 14%
Class 1: 25%
Class 2: 65%
Class 3: 10%
Class 1: 5%
Class 2: 90%
Class 3: 5%
Average:
Class 1: 26.2%
Class 2: 65.6%
Class 3: 8.2%
Final prediction: Class 2