# Assignment #3

**Course**: ISA 414/514

**Instructor**: Dr. Arthur Carvalho

**Points**:100

**Due date**: October 7th, 2022, before 11:59 pm

**Submission instructions**: this assignment is to be done individually. All your answers should be in a single *Python* notebook. Your code must be well formulated (*i.e.*, no errors) and sound (*i.e.*, it does what the question asks it to do). In particular, the grader must be able to open your *.ipynb* file using VS Code and run the code without running into errors. Code with errors may receive zero points. Submit the final document on Canvas before the due date.

**Question 1:** suppose you develop credit score systems for the famous Carvalho World Bank. Your current task is to develop a system that predicts whether a future client asking for a loan will default on payments, *i.e.*, to predict whether a client is "good" or "bad." The system development follows the CRISP-DM methodology. When in production, this system will support the decision made by bank employees on whether or not to give a loan to a certain client.

Carvalho World Bank has several document-oriented databases in production, storing every single transaction of its clients. Obviously, the seasoned, world-class IT professionals working for Carvalho World Bank will not allow the data analytics/development team to query the databases in production. Instead, they created a separate database to be used exclusively by the analytics/development team. This separate database periodically receives clean data from the databases in production. To access such a database, you must connect to a MongoDB collection called "bank" inside a database called "ISA414" as a guest user. The guest user has only read access, and the URL required for the connection is:

**mongodb://guest:abcd1234@cluster0-shard-00-00.3wrhn.mongodb.net:27017,cluster0-shard-00-01.3wrhn.mongodb.net:27017,cluster0-shard-00-02.3wrhn.mongodb.net:27017/?ssl=true&replicaSet=Cluster0-shard-0&authSource=admin&retryWrites=true&w=majority**

a) **Data collection [10 points]:** from inside your notebook, collect the required data by retrieving everything from the bank collection and saving the result to a data frame called bank_data. Hint: say the variable result stores the results from your query to obtain all the documents from the database. You can convert result to a list as follows results = list(results). Then, you can obtain the data frame bank_data as follows: bank_data = pandas.DataFrame(results)

b) **Data preprocessing [20 points]:** we shall use the module sklearn in our endeavors. Recall that models in sklearn do not accept qualitative variables. Other than target, which will be our target variable to be predicted, create dummies for all the qualitative variables in the data frame bank_data (technically, this may not be the best encoding since some variables are ordinal). Make sure you remove one redundant dummy variable for each original variable. Next, recode target as a 0/1 variable. Once you are done, randomly split bank_data. The resulting training data must contain 660 observations, and the test data must have 340 observations. Important note: look at the data dictionary for the data types. Do not blindly trust Python picking up the right attribute types for you.

c) **Modeling [10 points]:** train two models using the training data, namely a decision tree and a random forest (1000 trees). Important note: would you enter the variable _id into your models?

d) **Evaluation [10 points]:** evaluate your models in the wrong way. In particular, predict the target values on the training data, and compare the predictions against the true target values in the training data. Report the overall accuracy of the decision tree and the random forest.

e) **Evaluation [10 points]:** evaluate your models correctly now. In particular, predict the target values on the test data, and compare the predictions against the true target values in the test data. Report the overall accuracy of the decision tree and the random forest.

f) **Reflection [10 points]:** what happens with the accuracy of both models when you evaluate them using the test data as opposed to the training data? Explain why this happens. Finally, which model is more accurate and, thus, more likely to be deployed in practice? Sloppy answers will receive 0 points.

**g) Deployment [20 points]:** Based on your decision-tree model, would you give a loan to the following person? You must manually create a data frame containing the values highlighted below, encode the values using the same OneHotEncoder you created in 1-b), and use the resulting data frame together with the predict function to get a response:

- The applicant has no checking account;
- The applicant is asking for a 72-month loan;
- His credit history says all paid;
- The purpose of the loan is to buy a new car;
- The credit amount requested is equal to 2015,
- The applicant has no known savings;
- The applicant is employed for more than 7 years (>=7);
- The installment commitment is defined as 4;
- The applicant is a male single;
- The applicant is the guarantor of other parties;
- The applicant has lived at the same address for 3 years;
- When it comes to property magnitude, the applicant has one car;
- The applicant's age is 43;
- The applicant has no other payment plans (none);
- The applicant rents a house (rent);
- The applicant has 2 existing credit lines;
- The applicant's job type is skilled;
- The applicant has one 1 dependent;
- The applicant owns a telephone (yes);
- The applicant is not a foreign worker (no);

**h) Accountability [10 points]:** for the sake of potential auditing, store the unstransformed (i.e., before applying encoders) data about the new client plus your model's prediction in your MongoDB database. That is, you have to create a new connection to your personal database created in Lecture 11. Specifically, create a database called "bank," a collection called "predictions," and store all the information about the client asking for a loan plus the model's result inside a single document. Hint: put all the information inside a data frame, transform the data frame to JSON, and send the result to Mongo.