Data Science - Final Individual Delivery Guide
July 2021 - The Bridge

INSTRUCTOR: Gabriel Vázquez Torres
gabriel@thebridgeschool.es

TEACHER: Leonardo Sánchez
leonardo@thebridgeschool.es

TEACHER: Borja Puig de la Bellacasa
borja@thebridgeschool.es

TEACHER: Clara Piniella Martínez

Delivery explanation

This individual delivery aims to practice different concepts about EDA, Machine Learning and APIs. Also, the project will be presented.

The student must choose a subject that he/she prefers. This project can be an extended version of the first individual project.

Requirements

The next requirements are mandatories:

1. The project must give an answer to a **hypothesis** (explained below) if there is an EDA part.
2. The program must give one or more predictions from a Machine or Deep Learning algorithm that make sense with the idea of the project.
3. The student will do a presentation explaining all steps he/she has done.
4. It is mandatory for the student to divide all steps the project has as the first step in this project.

5. It is mandatory for the student to use *trello* (or other related) to manage the tasks in different states: TODO, DOING and DONE. BACKLOG and REVIEW are optionals.
6. The delivery must be sent before 11/07/2021 at 23:59. The presentation will be 12/07/2021 and/or 13/07/2021.
7. The delivery must be sent in a *.zip* file by email with this structure:
    a. A folder **src/** that contains all the source code.
    b. A folder **documentation/** that contains all the documents related to documentation.
    c. A folder **resources/** that contains other useful content (images,...)
    d. A folder **models/** that contains trained and serialized models and models information.
    e. A folder **reports/** that contains all related to created reports such as figures, html, pdf, etc
    f. A folder **data/** that contains the data of the project (optional).
    g. A folder **notebooks/** that contains notebooks for your tests.
    h. A folder **src/utils/** that contains all the modules used by the *main* file.
    i. A file **src/main.ipynb** that contains all the functionality. This file must only contain imports, pandas, matplotlib, requests,... and calls to your **src/utils/*** modules.
    j. A folder **src/dashboard/** that contains an "*app.py*" file that will run a streamlit dashboard using the necessary command. There is, at least, these menu sections:
        i. "Welcome": here the user must see some information about the project and your profile.

ii. “Visualization”: here the user must see some graphs/images of your data.

iii. “Json API-Flask”: here the user must see a table/dataframe from your cleaned data. Optional if you do not have dataframe data.

iv. “Model Prediction”: here the user must see model information and must be able to execute a prediction using your saved model. If the prediction must be done with text, then the user must be able to write text. If the prediction must be done with an image/file, then the user must be able to upload an image/file.

v. “Models From SQL Database”: here the user will see the models comparison from MySQL. The table of comparison is called *“model_comparasion”*. Its columns are: [“model”, “parameters”, “recall”, “score”] or [“model”, “parameters”, “rmse”, “r2”]

vi. Others you need. For example, the table “predictions” if you have it.

k. There are, at least, four modules inside **src/utils/** :

i. “folders_tb.py” that contains the generic functionality related to open, create, read and write files.

ii. “visualization_tb.py” that contains the generic functionality related to pandas, matplotlib, seaborn and other libraries focus on visualizations.

iii. “mining_data_tb.py” that contains the generic functionality related to collect data, clean data and others (wrangling methods such us working with multiples jsons)

iv. “apis_tb.py” that contains the generic functionality related to working with APIs.

v. “models.py ” that contains functionality used in the ML/DL section.

vi. “dashboard_tb.py” that contains the functionality related to the dashboard.

vii. “sql_tb” that contains the functionality related to SQL.

viii. Others that the student needs.

l. A file **src/api/server.py** that contains the functionality that starts the Flask API. This file only will be executed if it is passed an argument *“-x ‘yourfirstname’ ” (argparse)*, otherwise will show “wrong password”. There is, at least, these GET functions:

i. One that must allow you to receive a ***token_id*** value and, if ***token_id*** is equal to S, return the jsons that contains the logic explained below Otherwise, return a string with a message of error.

1. S is the DNI of student starting with the letter: Example: “B80070012”

ii. Another one that allows you to make some predictions using the saved model (**Option A**)

iii. A function that inserts your cleaned data (or metadata) to your MySQL database.

iv. Others that are relevant for the project.

m. The jsons that are returned are:

i. A json with the prediction done. The prediction must be coherent with the goal of the ML part. (**Option A**)

ii. Depending on the data and the problem, the student must return interesting data with the goal to make its program useful.

iii. A json that represents the data treated and cleaned.

Hypothesis

Normally, the goal in an EDA project is answering a question or demonstrating an axiom. This is, giving all necessary reasons to explain why the answer to the question is *one specifically* and refute or reaffirm an axiom.

One example of a hypothesis in the project of covid-19 could be:

> *We believe that the alarm state of each country has an impact on the progression of daily infection.*

Maybe, for your project, there is no possible hypothesis.

Presentation

All students can do a presentation about its project. The presenter will use a presentation file (no PDF or code directly) to explain all the steps of the workflow with graphs.

The duration of the presentation won't be longer than 10 minutes so it is really important and necessary to explain the essential points of the work.

This time, the judge will stop the presentation after 10 minutes.

The project steps

The idea of the project consist in different steps:

1. Find the subject: the student must find the project itself or use the last one if it has enough data. This is something he/she wants to do.
2. Find the data related to the project: research where it can be and if it is accessible from the public.
3. Define a hypothesis: find something you can conclude with your data. (optional)
4. Define the necessary steps to demonstrate or not your hypothesis. (optional)
5. With the code structure defined and using Python:
    a. Get your data. Maybe you need to use an API, maybe a file. Data Wrangling.
    b. Clean your data. Detects outliers, rare values and reemplace NaN values if needed.

c. Draw all graphs you need both to understand your data and to show the necessary results.
d. Create an API that returns the explained in the **Requirements** section. Maybe you find it useful doing more than the two endpoints explained.
e. Create the Streamlit Dashboard.
f. Explain why from your graphs and others results it can be argued the conclusion (if any).
g. Describe why it is useful the predictions you will do with your AI model.

6. Document all steps, zip the necessary files, send it to teachers' mails and upload it to classroom. If you have huge data, upload it to wetransfer and attach the link.

NOTE: Do all steps finishing the criteria requirements.

The resources

With the goal of finding all the necessary resources, the student can search all over the internet.

There are pages where you can find both good examples of EDA projects and datasets:

- Kaggle: here you can find millions of examples with millions of datasets. There are different parts where you can learn from novices or experts.
- Aicrowd
- Paperswithcode
- Googledatasetsearch: here you can find millions of datasets. It is a good page if you want to find the data you need.
- GoogleApis: here you have many apis from different subjects to get data.
- City council pages, statistics pages and thousands of APIs you can find on the Internet.
- Statista
- https://data.world/
- https://ourworldindata.org/
- http://www.fao.org/statistics/databases/es/
- https://informationisbeautiful.net/data/

- https://archive.ics.uci.edu/ml/datasets.php?format=&task=reg&att=&area=&numAtt=&numIns=&type=&sort=nameUp&view=table
- https://www.gapminder.org/data/
- Other ones in EDA delivery part 1 and Discord.
- Add the words: API or CSV or JSON when you search on Google

If the student has not any inspiration, then we can recommend the next subjects:

- Analyzing how the pandemic situation has changed lives in some subjects. Predict the future of the pandemic.
- Analyzing tweets to determine if some event changes the tendencies. Analyze the sentiments and classify new tweets with some feelings.
- Analyzing films datasets to conclude if there are more female actresses or romantic films. Predict the profit of a new film from its properties.
- Analyzing sport datasets to conclude if Messi, LeBron James or Fernando Alonso are the best in their sports. Predicting how many goals will score Messi next season.
- Analyzing disease datasets to conclude if there are relationships between symptoms and number of deaths (or other relationship). Predict if a new patient will survive from some symptoms.
- Analyzing climate datasets to conclude if climate change is real. Predict the future of the climate (forecasting).
- Analyzing video datasets to conclude if the funny videos have the most views. Predict the views or likes of a video from its duration and genre.
- Training images and classify new images from different subjects: medicine, animals, signals, etc

  https://www.kaggle.com/meowmeowmeowmeowmeow/gtsrb-german-traffic-sign/

- Detecting faces and predicting the feelings of a person.
- Analyzing text and predicting the feeling of the text (Natural Language Processing).
- Predicting the future of a time series (forecasting).
- Using openai (or other) to create a reinforcement learning project.
- Using GANs for generative modeling. Examples:

https://machinelearningmastery.com/impressive-applications-of-generative-adversarial-networks/

Evaluation criteria

For this delivery, there are different delivery options. Each student must choose what delivery they want to do. **C** is the minimum requirement for this delivery. There is a hierarchy in the options: **C→ B→ A→ A+***

It is not allowed to do:

- B without C
- A without B and C
- A+ without A, B and C

It is important to understand that in an entire Data Science project and due to the great variety of possible projects, these steps are the minimum you can do. It is obvious that the student will do a lot of more tasks that are described here.

In this type of project we assume that the student will learn more things that we have seen in class. And that is the goal.

When the student finishes the project, the student will have learnt more things than any other. For that, we recommend defining the steps the best he/she can do.

We understand that, maybe, the different questions below asked are irrelevant for your project. Try to complete those questions with coherent ones that make sense with your project.

You can do almost any Data Science project. You are free to choose your project and the complexity of it. The more complex you do your project, the more you will understand your project. Remember the goal, find a job.

In this project with the Machine Learning section there are thousands of projects you can do. It is not possible to see all the possibilities in class. In each project, you will learn something new that no one class in the world can teach you directly. That is the part of research.

Option C

Apart from all requirements that are written in the **Requirements** section, there are the next mandatory exercises:

1. Document all steps. Structure your code to keep it cleaned using good practices.
2. Collect the data. Try to do each call, it collects the last updated data.
3. Determine and explain if the data is cleaned. If not, then clean it.
4. Use Data Augmentation, dimensionality reduction and/or other techniques you need.
5. Create an API that returns a Json with the logic explained. The flask server must be executed running the **src/api/server.py** file.
6. Create the Streamlit dashboard.
7. Show different tendencies for each column in your dataset.
8. Represent, in a pie chart, the time you needed for each point in the **The project steps** section.
9. Choose the target (y) of your data. Which is the best model and hyperparams for the training? Create a model comparison using, at least, 5 different models with different hyperparameters.
10. Represent in a dataframe the top 5 models with their hyperparameters for your data.
11. Use your dataframe cleaned to insert its values to a MySQL database. The info of the database will be in the *"sql"* channel on Discord. To do that, you will need to create a Table with the dataframe columns. The name of the table is your entire name using "_" between words. Example: *"pepito_perez_galdos".* It is not allowed to upload more than 20k rows. If you have other data types, then upload the info of your dataset with these columns: ["id", "filename", "metadata"]. "metadata" column is related to file data, for example, resolution, filesize, etc
12. Answer the questions:
    a. Was it possible to demonstrate the hypothesis? Why? (optional)
    b. What can you conclude about your data study?
    c. What would you change if you needed to do another DS project?
    d. What do you learn doing this project?

Option B

1. Show the histogram of each column of your dataset with *bins*=5. How are the ranges painted?
2. Which are the columns with the highest correlation? Draw the correlation matrix.
3. Use Matplotlib functions to show all graphs. No with pandas directly.
4. The prediction of requirement **7.iv** will be inserted in a MySQL table with the model information. Table name: “predictions”, columns: [“model”, “parameters”, “prediction”]
5. During the evaluation step (when you are testing your trained model), show the predicted data with the real labels in some graphs in order to see the performance of your model visually.

Option A

1. Research to save each plot in local files.
2. Use distributed modules for each functionality. The jupyter notebooks must not have any loop or functions. It only must have the initial imports and the call to necessary functions.
3. Apart from matplotlib, use seaborn to show the graphs.
4. Show, visually, the progression of your training and validation score.
5. Create the gantt diagram of your project from the beginning and measure the time of your tasks. Attach two screenshots of the beginning and end of the project.
6. Answer the questions:
    a. Are there outliers or some rare data?
    b. What are the columns that have more repeated values?

Option A+

There are different A+. You can do the ones you want:

1. Create a pull request for the entire project.
2. How can you put your flask server with a public IP without Heroku? realize that flask starts the server in a private net as default (localhost)
3. How can you put your flask server with a public URL without Heroku?
4. How can you put your flask server with a public IP with Heroku?
5. How can you put your flask server with a public URL with Heroku?
6. Are there more urls from where to collect your data?. Explain why. If yes, then collect it and merge it with your data.
7. In order to practice OOP and engineering/architecture concepts in computing, define all the functions inside classes and make the program functional using them. After that, use a program to create the class diagram.
8. Using your own API url, use web scraping to get the json and show the data.
9. Save the features layers of your networks in order to do transfer learning.
10. Create a docker image with the capacity to mount your trained model in a container and be accessible from a port.
11. Create an EC2 machine that can be accessible from the internet and can run your api web from an IP and PORT.
12. Do the 12 but with the docker image in 11.
13. Manage your project with the CI/CD logic learned during the bootcamp.