

# PORTFOLIO



**Hogeschool van Amsterdam**

**Anh Thu Bui**

**Studentnumber: 500908730**

**HBO-ICT: Thematic semester Big Data WS 22/23**

## Table of Contents

|                         |    |
|-------------------------|----|
| Introduction .....      | 1  |
| Machine Learning.....   | 1  |
| Coding .....            | 3  |
| Data Collecting .....   | 4  |
| Data Cleaning .....     | 5  |
| Operations.....         | 7  |
| ML Model Building ..... | 8  |
| Data Visualization..... | 9  |
| Appendix .....          | 13 |

## Introduction

For the winter semester 22/23 I chose the minor 'Big Data' for my exchange semester. In today's everyday life, Big Data is becoming a more and more important and present topic and because of that, I think it is essential, especially for people who want to work in the IT sector, that they know at least the basics on how to use Big Data and what chances it holds. In addition to that, I am very keen to learn more about Machine Learning, Data collecting and analyzing, since I am thinking about going into this direction after my studies at university. Because of that, I think preferring the 'Big Data' minor over others such as Mobile Development or Internet of Things will help me reaching my learning goals.

## Machine Learning

**Current status: 19.10.2022**

While working on the onboarding assignment, I implemented two different Machine Learning algorithms: a logistic regression model and a naïve Bayes model to perform a sentiment analysis. Since the topic "Machine Learning" in the rubric requires the knowledge of at least three different machine learning algorithms, I did research about neural networks, how they work and how to implement them. I attended the workshop "Neural Networks" held on Monday on week 6 and also followed another tutorial to build a neural network, or to be specific, a Bidirectional LSTM (long short-term memory) model and trained it with the same twitter data provided for the onboarding assignment. A code snippet can be found under the chapter "Coding".

To underline that I researched about at least three different ML algorithms, I stated some pros and cons about logistic regression, naïve bayes and neural networks and stated some keywords about neural networks in general as well as bidirectional LSTM models.

## Logistic Regression

| Pro  | Cons  |
|--|---|
| Easy to implement, interpret and efficient to train  | Can lead to overfitting on the training set, when dataset is high dimensional. This happens when the model is trained on a small train data set with many features. This can be prevented with regularization techniques, which on the other hand, make the model complex. Adding high regularization techniques can also lead to underfitting. |
| trained parameters (weights) give inference about the importance of each feature<br>→ Relationship between features can be analyzed  | Difficult to capture complex relationships. More complex algorithms such as NNs outperform linear regression.   |
| Models can be updated easily to reflect new data (by using e.g. stochastic gradient descent)   | The model can make incorrect predictions when it is trained with features which are not important or relevant   |
| Outputs probabilities as well as classification results, which can be an advantage over models which can only give final classification results. E.g. the probability for one class can be 95% for one example, but also 55% for another. We get |   |

|   |   |
|---|---|
| more information on which examples are more accurate  |   |
| Very efficient when dataset has features that are linearly separable                              | Non linear problems cannot be solved, since it has a linear decision surface. Usually, in real world scenarios data is also not linearly separable. |
| Training time is far less than most complex algorithms due to simple probabilistic interpretation |   |

## Naive Bayes

| Pros  | Cons   |
|---|--|
| Very fast, easy to implement  | If test data has a categorical variable that was not present in the training set, the model will predict zero as probability and will not be able to make any predictions for that class |
| Multi class prediction is possible  | The model assumes that all features are independent. In real life features are rarely independent  |
| Performs better than other models with less training data if assumption of independence of each feature holds |  |

## Neural Networks (general)

| Pros  | Cons  |
|---|---|
| Flexible, can be used for regression and classification problems                              | Black boxes, we do not know how much each independent variable influences dependent variables |
| Any data which can be made numeric can be used as training data set                           | Computationally expensive and time consuming to train   |
| Can handle nonlinear data with large number of inputs such as images containing many features | Need a lot of training data (data dependency)   |
| Predictions are fast once learned   |   |
| Can be trained with any number of inputs and layers (extremely flexible)                      |   |

## RNN (recurrent neural network)

- usually used in natural language processing
- Remember the sequence of data and use data patterns to give prediction
- Uses feedback loops, which allows sharing the data to different nodes and predictions according to the information (like a memory)
- Converts an independent variable to a dependent variable for its next layer

## Bidirectional LSTM Networks

- Sequence information can go into both directions (backwards future to past) or forward (past to future) to preserve the future and the past information

### Sources

<https://analyticsindiamag.com/complete-guide-to-bidirectional-lstm-with-python-codes/>

<https://www.upgrad.com/blog/naive-bayes-classifier/>

<https://iq.opengenus.org/advantages-and-disadvantages-of-logistic-regression/>

## Coding

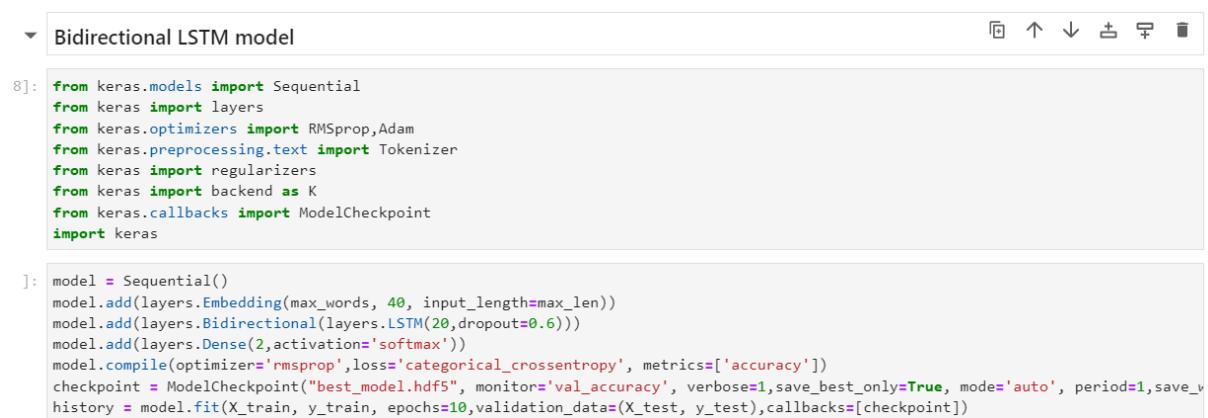
**Current status: 17.10.2022**

**Level: Good/Excellent (?)**

Additional to Marginal: The student has done coding for one of the following aspects

- Data import/ cleaning /storage
- ML algorithm
- Visualization with Python and can explain all produced lines of code

While working on our Big Data Project, I was responsible for the data collection part and used an API and Selenium, a python library for webscraping, to collect stock data and additional, general information on specific companies and stored them temporarily in a csv file (see chapter 'Data Collecting' for code snippets). In addition, I cleaned the data, removed unnecessary punctuation and html code indicating a new paragraph (see topic 'Data Cleaning'). I also did some research on another machine learning algorithm (Neural Networks) by attending the online workshop about neural networks and then implement my own for the sentiment analysis of the onboarding assignment (see chapter 'Machine Learning'). A code snippet of the neural network I created is shown below.



```
8]: from keras.models import Sequential
from keras import layers
from keras.optimizers import RMSprop, Adam
from keras.preprocessing.text import Tokenizer
from keras import regularizers
from keras import backend as K
from keras.callbacks import ModelCheckpoint
import keras

]: model = Sequential()
model.add(layers.Embedding(max_words, 40, input_length=max_len))
model.add(layers.Bidirectional(layers.LSTM(20, dropout=0.6)))
model.add(layers.Dense(2, activation='softmax'))
model.compile(optimizer='rmsprop', loss='categorical_crossentropy', metrics=['accuracy'])
checkpoint = ModelCheckpoint("best_model.hdf5", monitor='val_accuracy', verbose=1, save_best_only=True, mode='auto', period=1, save_weights_only=True)
history = model.fit(X_train, y_train, epochs=10, validation_data=(X_test, y_test), callbacks=[checkpoint])
```

Figure 1: Bidirectional LSTM model

```
[16]: test_loss, test_acc = model.evaluate(X_test, y_test, verbose=2)
print('Model accuracy: ', test_acc)

12357/12357 - 159s - loss: 0.4871 - accuracy: 0.7660 - 159s/epoch - 13ms/step
Model accuracy: 0.7659772038459778
```

Figure 2: Accuracy of bidirectional LSTM model

## Data Collecting

**Current status: 17.10.2022**

**Level: Good**

For the first sprint I was responsible for the data collecting part, which included stock data and other information such as KPI-values of different companies (Apple, Amazon etc.) and the most recent company news. In order to get the data, I used an API of the webpage “finance.yahoo.com” to write a script including methods to collect the data we want to process. Some code snippets can be found below:

```
...
Get performance index
Start and end date must be in the format YYYY-MM-DD
...
def get_OHLCV_data(self, start="", end=""):
    print("OHLCV History: ")
    # get data of last six months if no start and end date is set
    if not start:
        company_history = self.ticker.history(period="6mo", actions=False, interval="1d")
    elif not end:
        company_history = self.ticker.history(start=start, end=self.today, interval="1d")
    else:
        company_history = self.ticker.history(start=start, end=end, interval="1d")
```

Figure 3: Method to get KPI values in a specific time frame

```
class DataCollection:

    """
    Set executable for driver and options
    Comment options while developing
    """
    def __init__(self):
        pd.set_option('display.max_columns', None)
        pd.set_option('display.max_colwidth', None)
        options = Options()
        options.headless = True
        options.add_argument("--window-size=1920,1200")
        self.driver = webdriver.Chrome(ChromeDriverManager().install(), options=options)
        self.today = datetime.today().strftime('%Y-%m-%d')

        # set companies array and fill it
        comp = Companies()
        comp.fill_companies()

        # set indices array and fill it
        indices = Indices()
        indices.fill_indices()

        self.companies = comp.companies
        self.indices = indices.indices
```

Figure 4: Data Collection class

Furthermore, we wanted to include sentiment analysis on the company news and classify them whether they are positive or negative. Since the API we used fetches only the url links to each news

article, I used the Python library “Selenium” to navigate to each web page, which included the news article, and scrape its content:

Therefore, we had to use different sources to feed our dataset.

```
"""
Returns first part of news article, not the whole article!!
"""

def get_news_article(self, link_to_article, cookies):
    self.driver.get(link_to_article)
    wait = WebDriverWait(self.driver, 15)
    if cookies:
        wait.until(EC.presence_of_element_located((By.NAME, "agree"))).click()
    wait.until(EC.presence_of_element_located((By.XPATH, "//div[contains(@class,'caas-body')]/p")))
    p_elements = self.driver.find_elements_by_xpath("//div[contains(@class,'caas-body')]/p")[:10]
    text = ""
    for element in p_elements:
        text = text + element.text + " "
    return text
```

Figure 5: Fetch content of news article

I also used BeautifulSoup to scrape reviews from the webpage “careerbliss” for each company. This was not planned for the project at the beginning, but I just added it, in case we wanted to include reviews in our application.

```
def get_reviews(self, url):
    reviews_dict = {"position": [], "location": [], "review": []}
    headers = {"user-agent": "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/106.0.0.0 Safari/537.36"}
    page = requests.get(url, headers=headers)
    print(page)

    soup = BeautifulSoup(page.content, 'lxml')
    elems = soup.findChildren(class_='company-reviews')[:10]
    for elem in elems:
        children = elem.findChildren()

        for child in children:
            if child.get("class"):
                classes = child.get("class")
                if child.get("class")[0] == 'header5' or child.get("class")[0] == "job-title":
                    position = child.getText()
                    reviews_dict["position"].append(position)
                elif child.get("class")[0] == "review" or child.get("class")[0] == "comments":
                    review = child.getText()
                    reviews_dict["review"].append(review)
                elif child.get("class")[0] == "header13 spotlight-date" or child.get("class")[0] == "header13":
                    location = child.getText()
                    reviews_dict["location"].append(location)
    return reviews_dict
```

Figure 6: Method to scrape reviews from careerbliss

## Data Cleaning

### Level: Marginal/Good

I would rate my level good, because I did some additional, advanced data cleaning on the twitter dataset of the onboarding. This includes e.g. removing all sorts of punctuation, links, replies to users, containing the username, as well as removing stopwords, to achieve a cleaner data set

```
[49]: words = stopwords.words('english')
words.append('im')
words_set = set(words)
tr_table = str.maketrans('', '', string.punctuation)
def cleanTweets(tweet):
    #remove mentions
    tweet = re.sub(r'@[A-Za-z0-9]+', '', tweet)
    # remove hashtags
    tweet = re.sub(r'#', '', tweet)
    # remove Links
    tweet = re.sub(r'https?:\/\/[\S+]', '', tweet)
    tweet = re.sub(r'http?:\/\/[\S+]', '', tweet)
    tweet = tweet.lower()
    # remove stop words
    text_tokens = word_tokenize(tweet)
    tokens_without_sw = [word for word in text_tokens if not word in words_set]
    new_tweet = (" ").join(tokens_without_sw)
    # remove punctuation
    new_tweet = new_tweet.translate(tr_table)
    # remove words with only 2 or less characters
    text_tokens = word_tokenize(new_tweet)
    tweet_with_words_longer_than_3 = [word for word in text_tokens if len(word) >=3]
    new_tweet = (" ").join(tweet_with_words_longer_than_3)
    if new_tweet == '' or new_tweet == ' ':
        return tweet
    else:
        return new_tweet
```

Figure 7: Removing special/additional characters and stopwords etc.

Furthermore, I cleaned another dataset, which I have scraped from a webpage (careerbliss.com) before and included reviews of employees of different companies such as Apple or Adobe. This data set did not need much data cleaning, but there were still some special characters such as “\n”, which indicated a new paragraph in html and quotation marks. In addition to that I removed the first part of the values in the location column to prepare the data to visualize in our application:

```
[1]: import pandas as pd
df = pd.read_csv('reviews.csv', index_col=False)
pd.set_option('display.max_columns', None)
pd.set_option('display.max_colwidth', None)

[2]: df = df[['position', 'location', 'review']]
df
```

|   | position                             | location                         | review   |
|---|--------------------------------------|----------------------------------|--|
| 0 | Senior Business Intelligence Analyst | Posted 2 years ago in Austin, TX | "I worked at Apple for many years. The opportunity to be part of a movement under the innovative leadership of Steve Jobs was priceless. It truly was a Think Different time for the organization with the sky as anyone's limit. I later returned to Apple and saw firsthand how it transformed into the highly successful machine it is today led by Tim Cook. The two CEOs took Apple down very different paths and their objectives were night and day apart. Whether it was Apple when the rebellion was accepted or today's Apple that's just what you expected, it was and still is a great place to work." |
| 1 | Apple Employee                       | \n in New York City, NY\n        | "Great job and a cool company to work for with many perks and benefits. The only con is that it's a very large company so culture changes with the department and you can easily fade and become another fly on the wall."   |
| 2 | Apple Employee                       | \n in Chicago, IL\n              | "Good work environment. I have worked in a junior position but learned a lot. Thanks to my team."  |
| 3 | Operations Specialist                | \n in Houston, TX\n              | "I've worked for Apple for over 3 years now. They are an excellent company to work for as their benefits and workstyle is top-notch in the business. The only drawback is the jump between retail and corporate is near impossible without the long commitment."   |
| 4 | Apple Employee                       | \n in Cupertino, CA\n            | "Big company and innovative.\nGood employee benefits."   |

Figure 8: Review data set before cleaning

|       |  |
|-------|--|
| [10]: | def clean_data(df):<br>df.replace(to_replace=[r'\n', '\n', ''], value=['', '', ''], regex=True, inplace=True)<br>df['location'] = df['location'].str.split('in', n=1).str.get(-1)<br>return df |
| [11]: | clean_data(df)   |

|   | position                             | location          | review   |
|---|--------------------------------------|-------------------|--|
| 0 | Senior Business Intelligence Analyst | Austin, TX        | I worked at Apple for many years. The opportunity to be part of a movement under the innovative leadership of Steve Jobs was priceless. It truly was a Think Different time for the organization with the sky as anyone's limit. I later returned to Apple and saw firsthand how it transformed into the highly successful machine it is today led by Tim Cook. The two CEOs took Apple down very different paths and their objectives were night and day apart. Whether it was Apple when the rebellion was accepted or today's Apple that's just what you expected, it was and still is a great place to work. |
| 1 | Apple Employee                       | New York City, NY | Great job and a cool company to work for with many perks and benefits. The only con is that it's a very large company so culture changes with the department and you can easily fade and become another fly on the wall.   |
| 2 | Apple Employee                       | Chicago, IL       | Good work environment. I have worked in a junior position but learned a lot. Thanks to my team.  |

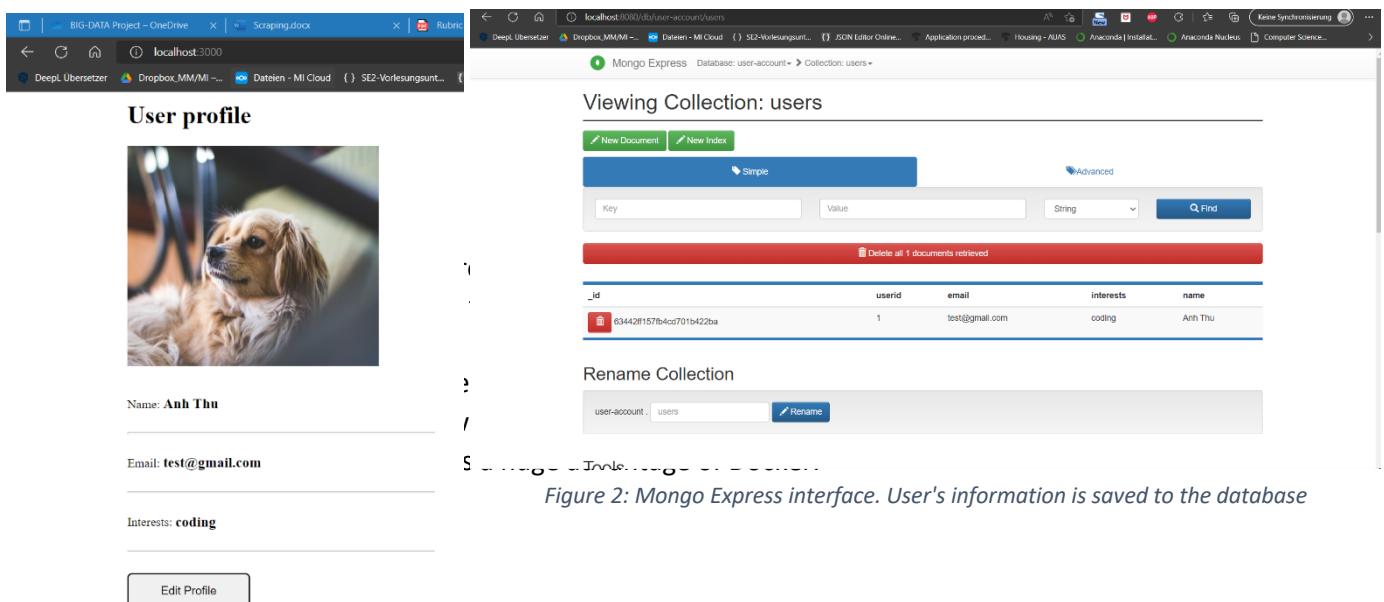
Figure 9: clean\_data method and data set after cleaning

## Operations

Current status: 08.10.2022

Level: Good

Since my goal for my first learning story was to learn more about docker, what advantages it holds and how to use it, I implemented a small project, where I included as many functions docker offers as possible. In that project, I followed a tutorial I found online and adjusted it to my needs. It contains a simple Javascript webpage, showing a form where the user can type in their name, email address and other information. By clicking on the button “Save”, all information is saved in the collection “users” of a MongoDB database.



The image shows a dual-screen setup. The left screen displays a web browser with a form for creating a user profile. The right screen shows the MongoDB Express interface, specifically the 'users' collection.

**User profile (Left Screen):**

- Name: Anh Thu
- Email: test@gmail.com
- Interests: coding

**Mongo Express (Right Screen):**

Viewing Collection: users

| _id                      | userid | email          | interests | name    |
|--------------------------|--------|----------------|-----------|---------|
| 63442ff157fb4cd701b422ba | 1      | test@gmail.com | coding    | Anh Thu |

Rename Collection: user-account | users | Rename

Figure 2: Mongo Express interface. User's information is saved to the database

Figure 1: Web page

```

! mongo.yaml
1  version: '3'
2  services:
3    my-app:
4      image: thuthuts/docker-app:1.1
5      ports:
6        - 3000:3000
7      restart: always
8    mongodb:
9      image: mongo
10     ports:
11       - 27017:27017
12     environment:
13       - MONGO_INITDB_ROOT_USERNAME=[REDACTED]
14       - MONGO_INITDB_ROOT_PASSWORD=[REDACTED]
15     volumes:
16       - mongo-data:/data/db
17    mongo-express:
18      image: mongo-express
19      ports:
20        - 8080:8081
21      restart: always
22      environment:
23        - ME_CONFIG_MONGODB_ADMINUSERNAME=[REDACTED]
24        - ME_CONFIG_MONGODB_ADMINPASSWORD=[REDACTED]
25        - ME_CONFIG_MONGODB_SERVER=mongodb
26    volumes:
27      mongo-data:
28        driver: local

```

Figure 3: Docker compose file

## ML Model Building

While working on the onboarding assignment, I added a hyperparameter tuning for both of the models I built for the machine learning part, which was not a requirement of the onboarding. In order to do this, I used RandomizedSearchCV instead of GridSearch. For both options, we can provide a list of hyperparameters and test, which one are the best to achieve the highest accuracy after training. The only difference of RandomizedSearchCV to GridSearch is, that not every combination of the provided list of hyperparameters is tried out, but only random combinations. The advantage of that is, that a wide range of values can be tested, but it is not guaranteed that the best parameter combinations are returned, since not every combination is used.

```

Tuning hyperparameters

kf = KFold(n_splits=5, shuffle=True, random_state=42)

param_grid = [
    {
        'penalty' : ['l1', 'l2'],
        'C' : np.logspace(-4, 4, 20),
        'solver' : ['liblinear']}
]

# Instantiate the RandomizedSearchCV object
logreg_cv = RandomizedSearchCV(lr, param_grid, cv=kf)

# Fit the data to the model
logreg_cv.fit(X_train, y_train)

# Print the tuned parameters and score
print("Tuned Logistic Regression Parameters: {}".format(logreg_cv.best_params_))
print("Tuned Logistic Regression Best Accuracy Score: {}".format(logreg_cv.best_score_))

```

Figure 10: Parameter tuning using RandomizedSearchCV

In the figure above, I created a small list of parameters used to find the best parameters for my logistic regression model. I added two different penalty options, as well as some values for the

hyperparameter C, which stands for the inverse of regularization strength and a solver, the algorithm to use in the optimization problem. To validate the model, I added a cross-fold validation with five folds.

After running the search, the following combination of parameters were returned:

```
Tuned Logistic Regression Parameters: {'solver': 'liblinear', 'penalty': 'l2', 'C': 0.23357214690901212}
Tuned Logistic Regression Best Accuracy Score: 0.7819959743882462
```

After that, I started another RandomizedSearchCV, but added more parameters to my grid:

```
kf = KFold(n_splits=5, shuffle=True, random_state=42)
params = {"penalty": ["l1", "l2"],
          "tol": np.linspace(0.0001, 1.0, 50),
          "C": np.linspace(0.1, 1.0, 50),
          "class_weight": ["balanced", {0:0.8, 1:0.2}]}
# Instantiate the RandomizedSearchCV object
logreg_cv = RandomizedSearchCV(lr, params, cv=kf)

# Fit the data to the model
logreg_cv.fit(X_train, y_train)

# Print the tuned parameters and score
print("Tuned Logistic Regression Parameters: {}".format(logreg_cv.best_params_))
print("Tuned Logistic Regression Best Accuracy Score: {}".format(logreg_cv.best_score_))
```

Figure 11: Second RandomizedSearch on Logistic Regression model

With this, much more combinations can be tested to find the best parameters for the model created.

## Data Visualization

Level: Good

Since working with Tableau in the module “Data Visualization” was very interesting and fun, I decided to create an additional dashboard as my own project for my portfolio, which was not part of the module “Data Visualization” or our big data project. I would rate my level as “Good”. Although I did not use Python or another programming language to create the visualization (since I did it with Tableau), I nevertheless followed the mantra “Overview first, zoom and filter, then details-on-demand” as good as possible in my dashboard. A first glance on the dashboard is shown below:

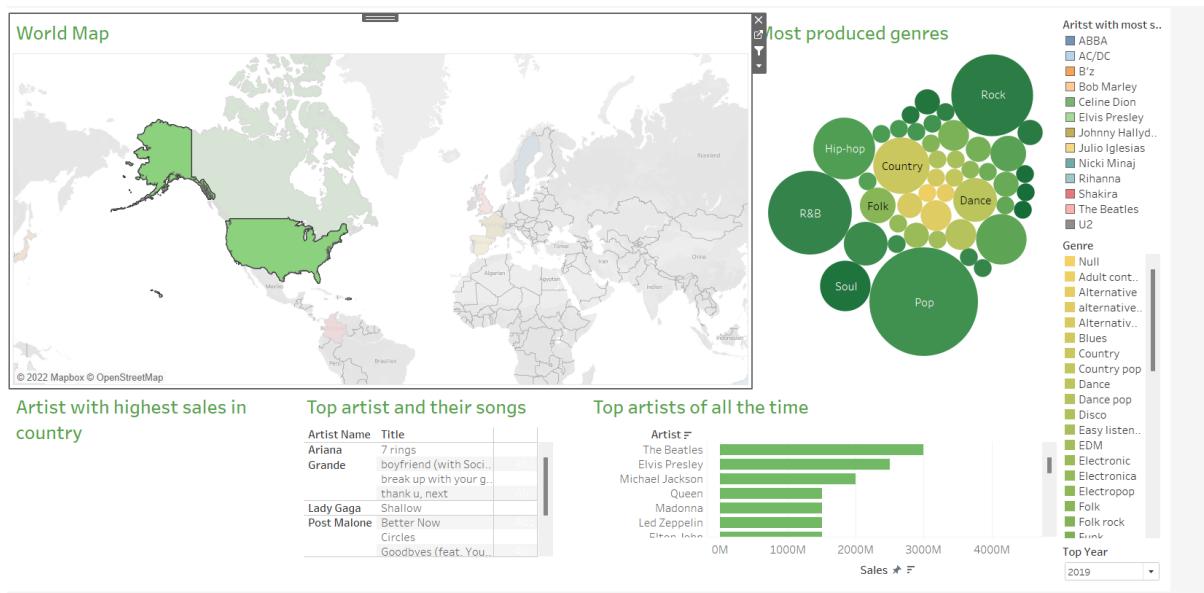


Figure 12: Dashboard

For this dashboard, I used two different datasets and combined them, which can be found under these links:

- <https://www.kaggle.com/datasets/kabhishm/best-selling-music-artists-of-all-time?resource=download>
- <https://www.kaggle.com/datasets/muhmores/spotify-top-100-songs-of-20152019>

Before actually using the datasets, I had to clean the datasets first and did some adjustments. E.g. did I have to add additional columns such as adding the artist with the highest sales of each country to the dataset “best selling music artists of all time”. To do this, I created a calculated field as shown below:

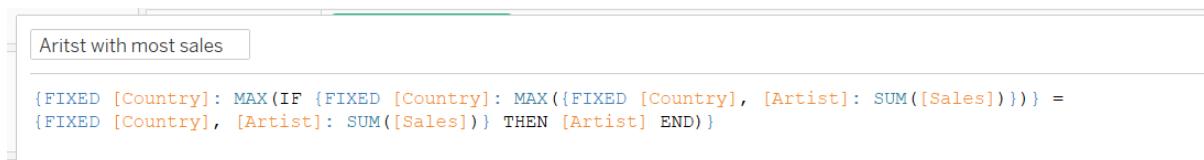


Figure 13: Formula to create Artist with most sales column

In addition to that, I had to correct spelling mistakes. An example for this would be the genre column, where some genres were written only in small letters and others correct with a starting capital letter. The problem here was that I had the same genre, but spelled differently, so I wrote another formula to fix the mistakes. Another aspect was that all the genres were written in the same column, divided by a “/”. Because of that, I had to split all values in that column so that an additional column is added for each genre. On top of that, I used the Pivot function in Tableau to transform the table and changed the added genre columns as rows, so that I would have an additional row of data with every genre. I had to do the same process with the country column, since it also contained multiple values. All these adjustments were necessary so that I could use them in my final dashboard.

When looking at the dashboard, the most prominent view is the world map. I placed it on the top left and resized it so that it is the largest view in the dashboard. The world map should show an overview first. All countries, which are listed in the dataset, are colored in a specific color, representing one

artist, which is first in the list of the artists with the most sales. The country is the country they come from.

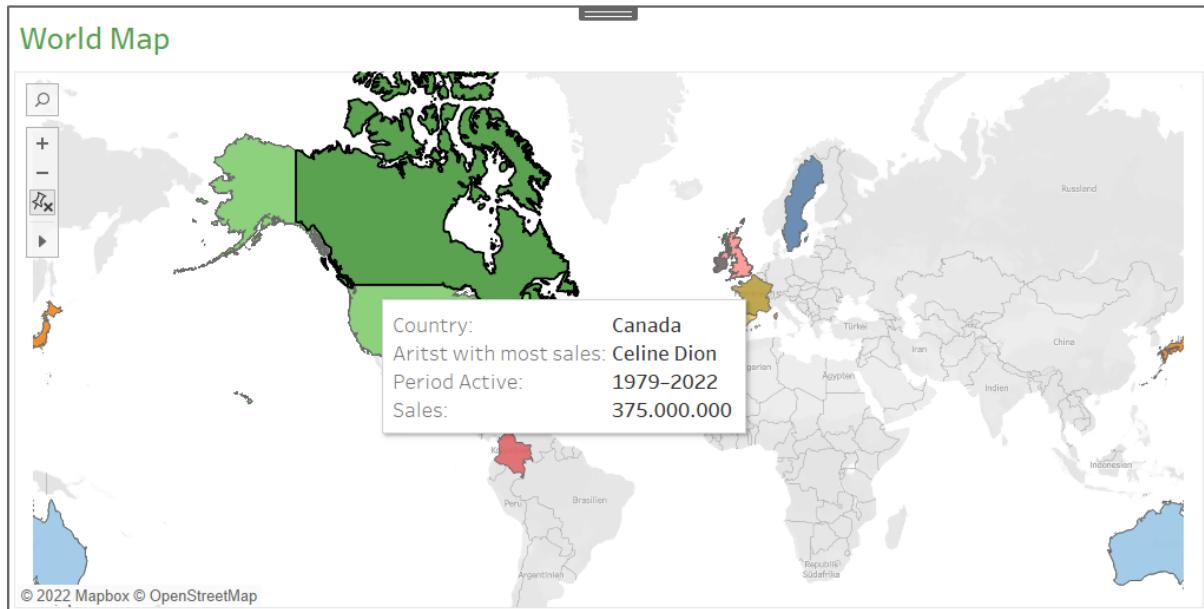


Figure 14: World Map

In the figure above one can see that not all countries are colored, since the dataset only contained artists, which had at least over 75 million record sales and most of them are from the U.S. By hovering over the countries, a tool tip appears showing details-on-demand such as the correlated artist, the period active and the total sales as well as the name of the country.

By clicking on a country in the world map, a filter is triggered and all other countries turn grey to underline that the focus is now on the selected country. Selecting a specific country, changes the view on the right:

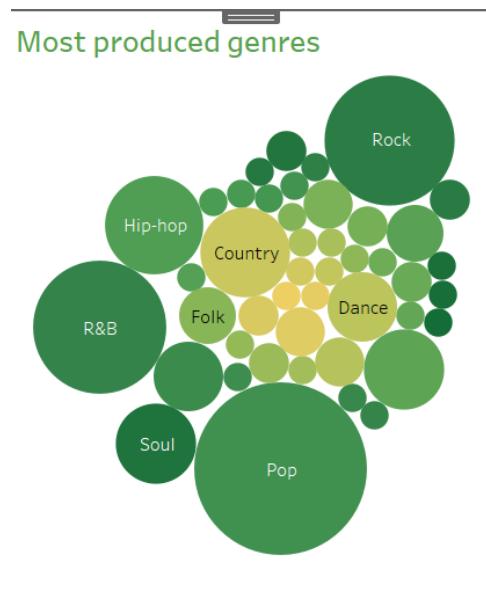


Figure 15: Bubble chart

The bubble chart has the same function as a world cloud and counts the number of genres, which were produced in that specific country. In the U.S. the most prominent genres are therefore Pop, R&B and Rock. Besides that, the view on the bottom left also changes.

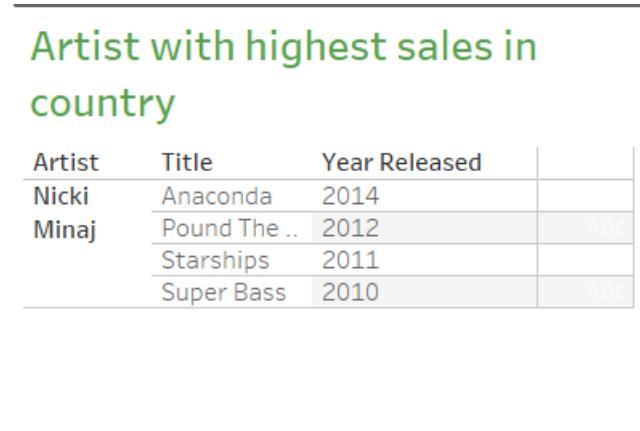


Figure 16: Artist with highest sales in country

In this view, the artist which had the highest sales in the selected country is shown with some of their most popular songs and their release dates. However, a problem here was that the datasets did not fit perfectly. The figure above shows the top artist for the country Trinidad. The artist with the highest sales in Canada e.g. was Celine Dion, but the other dataset, which contained the songs of the artists, did not have Celine Dion as artist saved, since it only contained the top 100 songs from 2015 to 2019. Because of the lack of information, I was also restricted in using other views such as line graphs etc. so I added only a table.

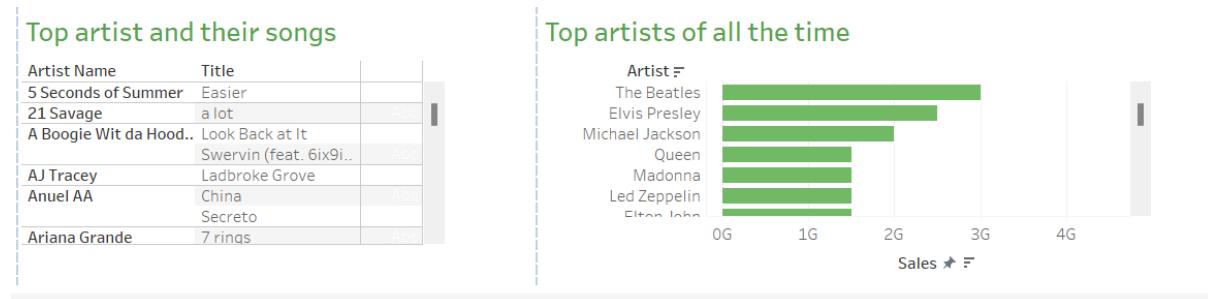


Figure 17: Top artist and their songs and top artists of all the time

The last two graphs are independent from the selection above. The table on the left shows all top artists along with their most popular songs in a specific year. The year can be selected from a dropdown menu on the left:



Figure 18: Dropdown filter

The last view shows a bar chart with the top artists of all the time and their sales. The user can click on a button above the bars, which appears when hovering over them, to sort them descending or ascending. The default setting is a descending order. Hovering over the bars also shows the exact sales value.

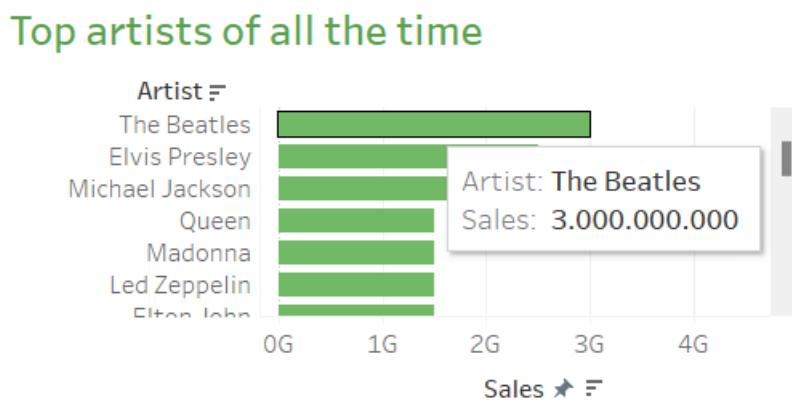


Figure 19: Details on demand in bar chart

The dashboard follows as I said, the mantra. The world map shows an overview first. The user can zoom in or out to explore the artists in that country as well as hover over them to get deeper insight provided by the tool tips. Besides, the user can filter the information by selecting a country or using the year filter on the right to only show information, which is relevant at the moment.

## Appendix

### [A1] Learning Journey

#### Learning Journey

In this semester where I participate in the Big Data Minor, one of my main goals is to deepen my knowledge, especially in Machine Learning, where I am most interested in and gain more experience and overall understanding of Big Data and its usings. Last semester, I was already able to contribute to a small project, where we implemented a neural network which could recognize different sounds (e. g. a dog barking, applause etc.) based on data my university provided us. I would like to follow up on that by taking part in a new project with a bigger team (we were only two last semester) and international students, which will be a challenge for me, since I am used to work on projects with people I know. Later in the working world, this will be a standard, which I need to get used to. Besides that, I want to really understand the different processes of Machine Learning, which models and processes would be suitable for our specific project and how they work. In addition, I want to learn more about data gathering, which means finding data for our model, knowing if its suitable for our model and realizing whether it is of high quality or not. Since the dataset was already provided for us last semester, I do not know which aspects to consider when gathering data. Therefore, cleaning and preprocessing data will also be a topic I want to learn more about.

### [A2] Rubric

On the following pages the BDSE rubric is shown. Cells, which are filled with **green**, are the goals I want to achieve at the end of the thematic semester. In addition, I underlined the cells in **yellow** to show, how I think I currently stand and where I improved.

|                                    | Insufficient   | Marginal   | Good   | Excellent   |
|------------------------------------|--|--|--|---|
| <b>General Concepts</b>            |  |  |  |   |
| 1. Machine Learning                | The student has no idea about different models to be used for data science                     | The student can explain the role of supervised and unsupervised machine learning during a ML implementation. Moreover the student can explain the differences between regression and classification. | Additional to Marginal:<br>The student knows the pro's and cons of at least 3 ML algorithms  | Additional to Good:<br>Student has done research on ML algorithms and can use arguments for using a particular one beyond the mandatory literature  |
| 2. Coding                          | The student has done no coding beyond the onboarding assignment.                               | The student has done some coding beyond the onboarding assignment and can explain some basic lines of coding.  | Additional to Marginal:<br>The student has done coding for one of the following aspects <ul style="list-style-type: none"> <li>• Data import/ cleaning /storage</li> <li>• ML algorithm</li> <li>• Visualization with Python</li> </ul> and can explain all produced lines of code | Additional to Good:<br>The student has done coding for at least 2 of the following aspects <ul style="list-style-type: none"> <li>• Data import/ cleaning /storage</li> <li>• ML algorithm</li> <li>• Visualization with Python</li> </ul> and can explain all produced lines of code |
| <b>Data Scientist and Engineer</b> |  |  |  |   |
| 3. Data collecting                 | The student uses only one (provided) dataset and has little or no understanding of its content | The student uses only one (provided) dataset and has full understanding of its content   | Additional to Marginal:<br>The student has combined several (external) datasets. The student knows how to get data   | Additional to Good:<br>The student has implemented jobs to gather data from an API or by webscraping continuously   |

|  | g of its content  |  | from an API or by webscraping  |   |
|--|---|--|--|---|
| 4. Data cleaning                                       | The student can not turn a real life datasets into a usable dataset for Machine Learning.         | The student as done some basic cleaning of a (provided) dataset, like set datatypes, label encoding and hot encoding | Additional to Marginal:<br>The student has done advanced cleaning on several datasets, like removing emoticons and stopwords, manipulating addresses and so on   | Additional to Good:<br>The student has used for instance complex regex expressions and lambda functions to clean the data   |
| 5. Data storage  | The student can not explain the differences between a SQL and a NOSQL database                    | The student can store datasets into a database and can argue the choice between a SQL and a NOSQL Database           | Additional to Marginal:<br>Student knows how to handle incremental uploads of data in the database<br><br>In case a SQL Database is used , the student can explain the schema of the database<br><br>In case a NOSQL database is used, the student can explain the nesting of JSON strings by referring to the most common use cases for searching | Additional to Good:<br>In case a SQL Database is used : for communication with the database a ORM framework is used<br><br>In case a NOSQL Database is used: All queries are parametrized |
| 6. ML Model building, beyond the onboarding assignment | The student cannot explain any of the different statements in the code used to build a classifier | Student can explain only the basic statements in the code behind a classifier  | Additional to Marginal:<br>Student knows how to explain all the ins and outs of the pieces of code involved. In particular how to succeed in improving the overall accuracy,   | Additional to Good:<br>Advanced tweaking of the parameters involved in the used classifiers has been used   |

|                  |   |  |  |   |
|------------------|---|--|--|---|
|                  |   |  | for instance by using GridSearch   |   |
| 7. Visualization | There is no dashboard at all. The student has not made any visualizations   | There is either <ul style="list-style-type: none"> <li>• A Python IDE with interactive plots</li> </ul> Or <ul style="list-style-type: none"> <li>• An out of the box package like Tableau or Power BI is used.</li> </ul> | Additional to Marginal:<br>Student had coded a dashboard in a modern programming language. In an interactive way the mantra of visualization is implemented: <ul style="list-style-type: none"> <li>• Overview first</li> <li>• Zoom and filter</li> <li>• Then details-on-demand</li> </ul> | Additional to Good:<br>Sophisticated interactive elements are implemented by coding   |
| 8. Operations    | The student has no idea of the processes involved in application development, testing, deployment and operations. | The student can explain the the role of git repositories and python environments   | Additional to Marginal:<br>The student can run a local docker container with some of the coding beyond the onboarding assignment locally. The docker can run everywhere  | Additional to Good:<br>The student has taken care of the whole devops pipeline. Whenever a change in the code is made and pushed to a git repo, a new image is being built and the image can run everywhere |

### **Block 1: Requirements to pass the first block of the semester and get a grading for BSDE Data Engineering**

All students needs to score at least

1. General, all aspects at least GOOD. So, Machine Learning and Coding need to be good
2. You'll need to score at least other 3 rows in the section marginal.

### **Block 1: Grading**

1. Failing the above requirements → 4
2. Passing the above requirements
  - a. By the bare minimum → 6
  - b. Additional to a. : one extra row GOOD → 7
  - c. Additional to b. : one extra row GOOD → 8

- d. Additional to c. : All *General Competences* EXCELLENT → 9
- e. All aspects through all requirements EXCELLENT → 10

### **Block 2: Requirements to pass the second block of the semester get a grading for BSDE Data Scientist**

All students needs to score at least

- 1. General, both General concepts Machine Learning and coding at least GOOD
- 2. You'll need to score at least
  - a. 2 other rows in the section GOOD
  - b. 2 other rows in the section MARGINAL

### **Block 2: Grading**

- 1. Failing the above requirements → 4
- 2. Passing the above requirements
  - a. By the bare minimum → 6
  - b. Additional to a. : one extra row GOOD → 7
  - c. Additional to b. : one extra row GOOD → 8
  - d. Additional to c. : All *General Competences* EXCELLENT → 9
  - e. All aspects through all requirements EXCELLENT → 10