Python for data analysis
YearpredictionMSD dataset

final project

Alice Pinault– Paul Coiffet

Esilv A4 / 2021-2022

Project objectives

In this project, we worked on the Million Song Dataset (MSD) which is is a freely-available collection of audio features and metadata for a million contemporary popular music tracks.

The aim of this project is to predict the release date of songs from audio features.

Songs are mostly western, commercial tracks ranging from 1922 to 2011 with a peak in the 2000's.

Because of the large range of the release dates and the large number of independant variables, we decided to predict decade instead of years.

Dataset description

It’s important to understand the dataset given to us before starting coding.

We have 3 types of attributes in our dataset :

targets : Year (range from 1922 to 2011)

TimbreAvg : 1 to 12

TimbreCov : 12 to 78

This makes a total of 90 attributes for 515345 rows.

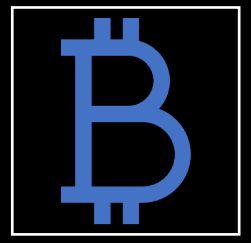
Features are extracted from the 'timbre' features from The Echo Nest API. We take the average and covariance over all 'segments', each segment being described by a 12-dimensional timbre vector.

Data pre-processing

Our first step was to clean the dataset and make it easier to work with.

We gave name to the columns, checked for nan values or null values and added our decade variable to prevent further plots and models.

Then we normalized with the min/max method the values in order to weight equally all the features.



We were now ready to work with this dataset even it was a bit too large (we noticed it later).

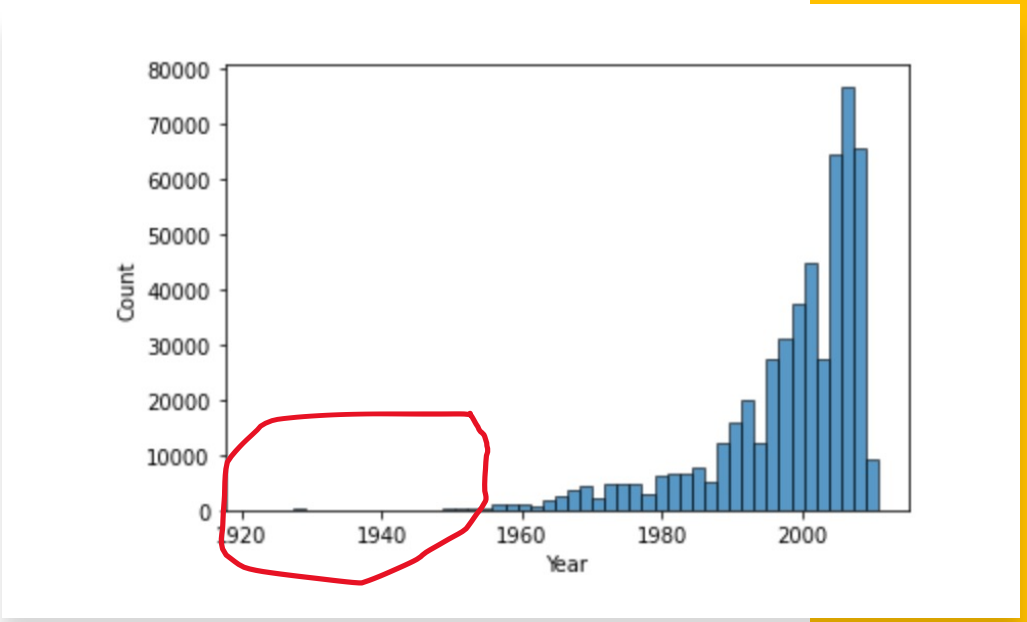
	Year	Decade	TimbreAvg1	TimbreAvg2	TimbreAvg3	TimbreAvg4	TimbreAvg5	TimbreAvg6	TimbreAvg7	TimbreAvg8	TimbreAvg9	TimbreAvg10	TimbreAvg11
0	2001	2000	0.800293	0.497205	0.599629	0.332545	0.370583	0.276962	0.452564	0.302498	0.492380	0.384046	0.462913
1	2001	2000	0.780177	0.492987	0.595220	0.341113	0.386532	0.229635	0.546232	0.359274	0.532466	0.453276	0.455925
2	2001	2000	0.817124	0.511606	0.571964	0.342073	0.394968	0.254988	0.512831	0.352096	0.522585	0.421927	0.459199
3	2001	2000	0.772129	0.464800	0.540674	0.319971	0.411973	0.224074	0.535929	0.311991	0.476686	0.345849	0.483895
4	2001	2000	0.817341	0.525963	0.590047	0.331972	0.374082	0.261984	0.487309	0.316833	0.509999	0.417425	0.452057

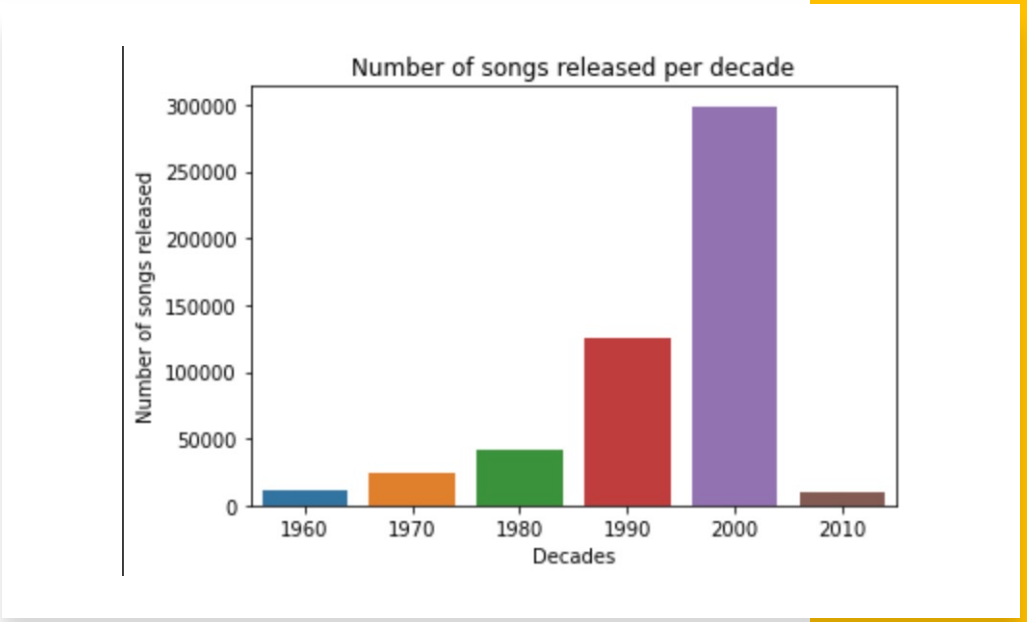
Data visualization

Our first idea was to check for the year distribution of songs.

We know from the dataset there is a peak in the 2000's, we've been able to confirm this:

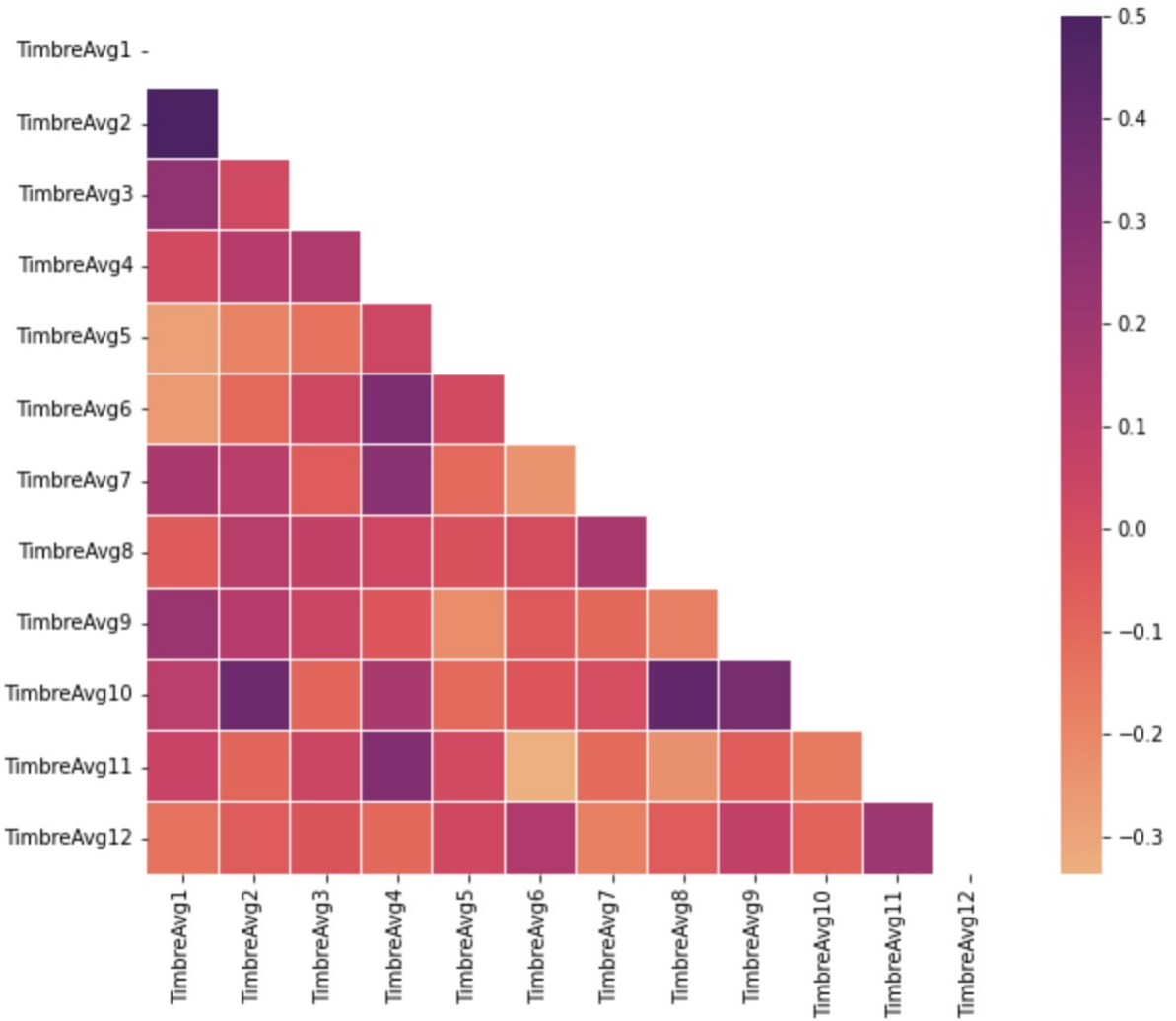
We noticed that there were not many songs released before 1960 thus, in order to have a more equal repartition of data through decades, we created a subset by deleting every instance below 1960.

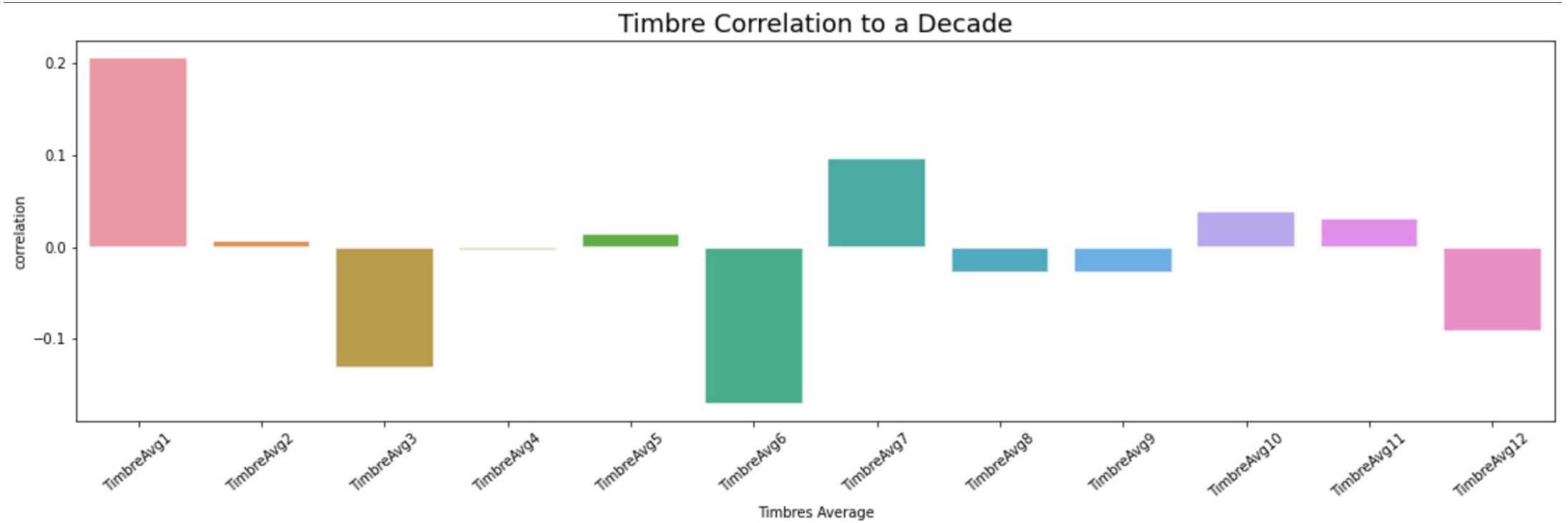




Correlation analysis 1

- We made an analysis of all the features through covariance.
- We started by checking the correlation between each average timbre. We used for this a heatmap :
- This shows us that there is indeed a correlation between the timbres but it's not a strong one. The maximum correlation is 0.56 between timbreavg1 and timbreavg2.





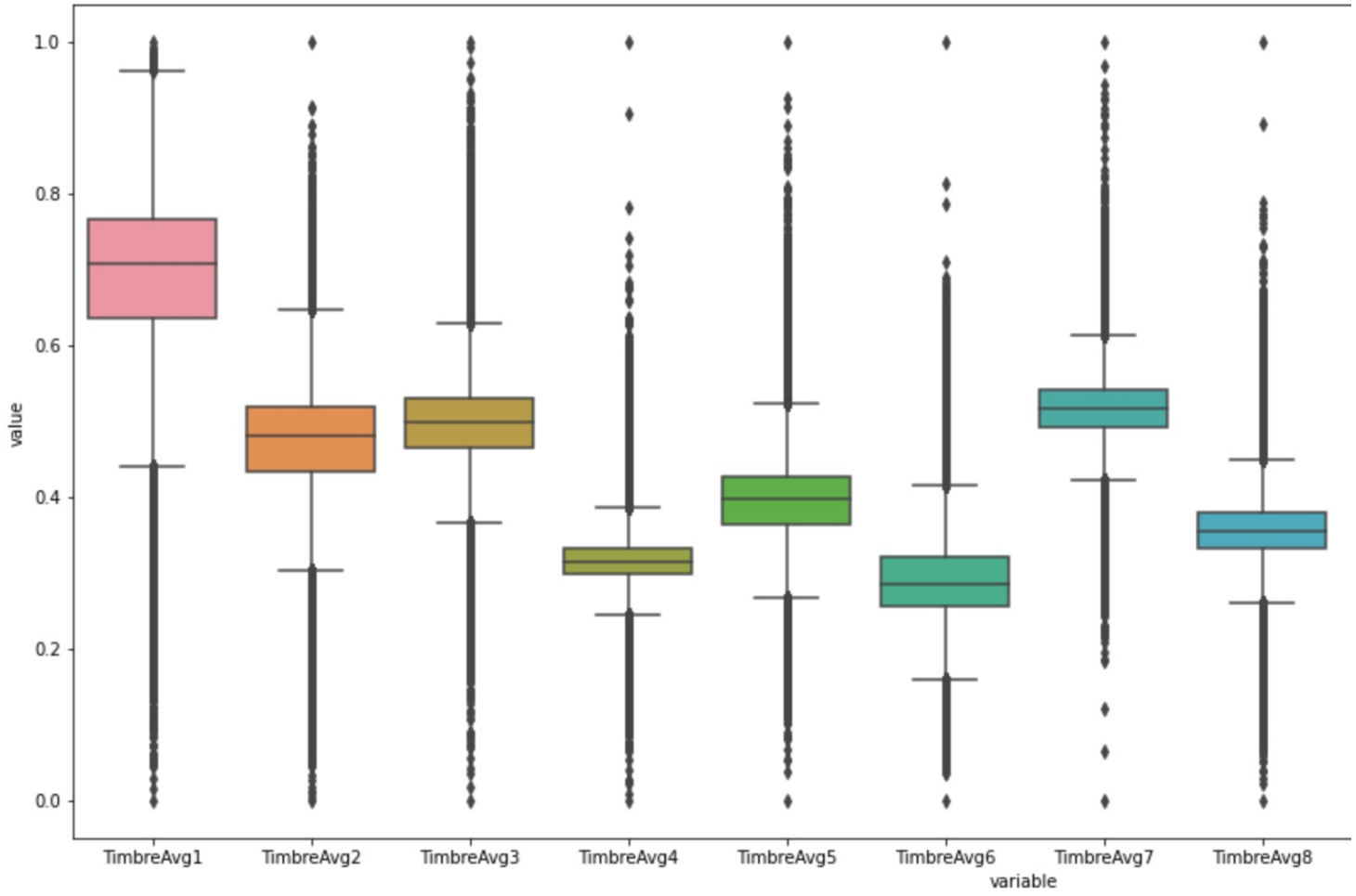
Correlation analysis 2

We then analyze the variation of a timbre to a decade. In this bar plot from Sns, we don’t notice any timbre having a major impact on a decade compared to an other.

The first and sixth timbre maybe stand out a little more but it’s not that representative.

Correlation analysis 3

- Finaly, we made an analysis of each timbre to all decades . We notice that all timbre are generally uniform and that there is not timbre standing out.



Data preparation

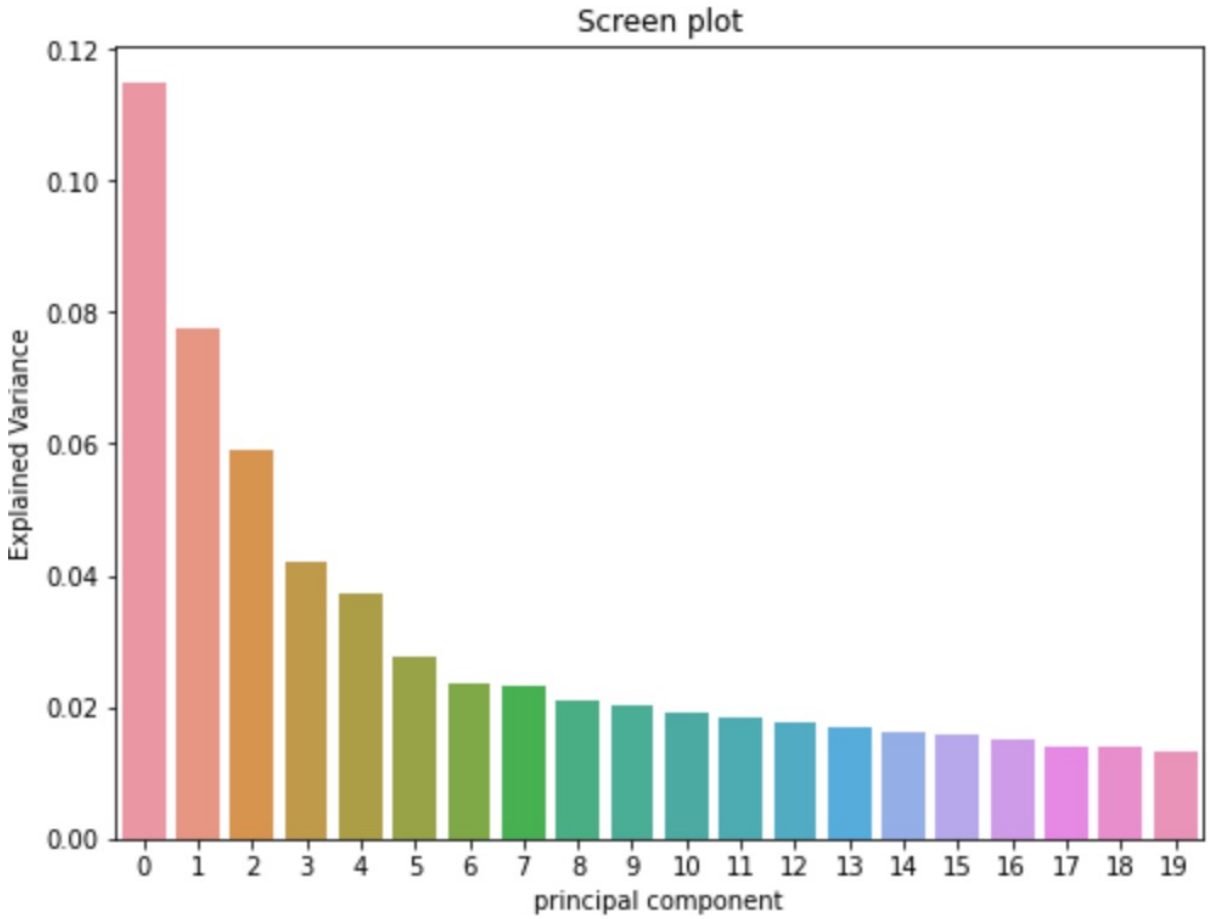
- To prepare our data, we used 3 methods :

- Min/max (that we already did)
- Data scaling
- PCA and T-SNE
- Downsize of the dataset

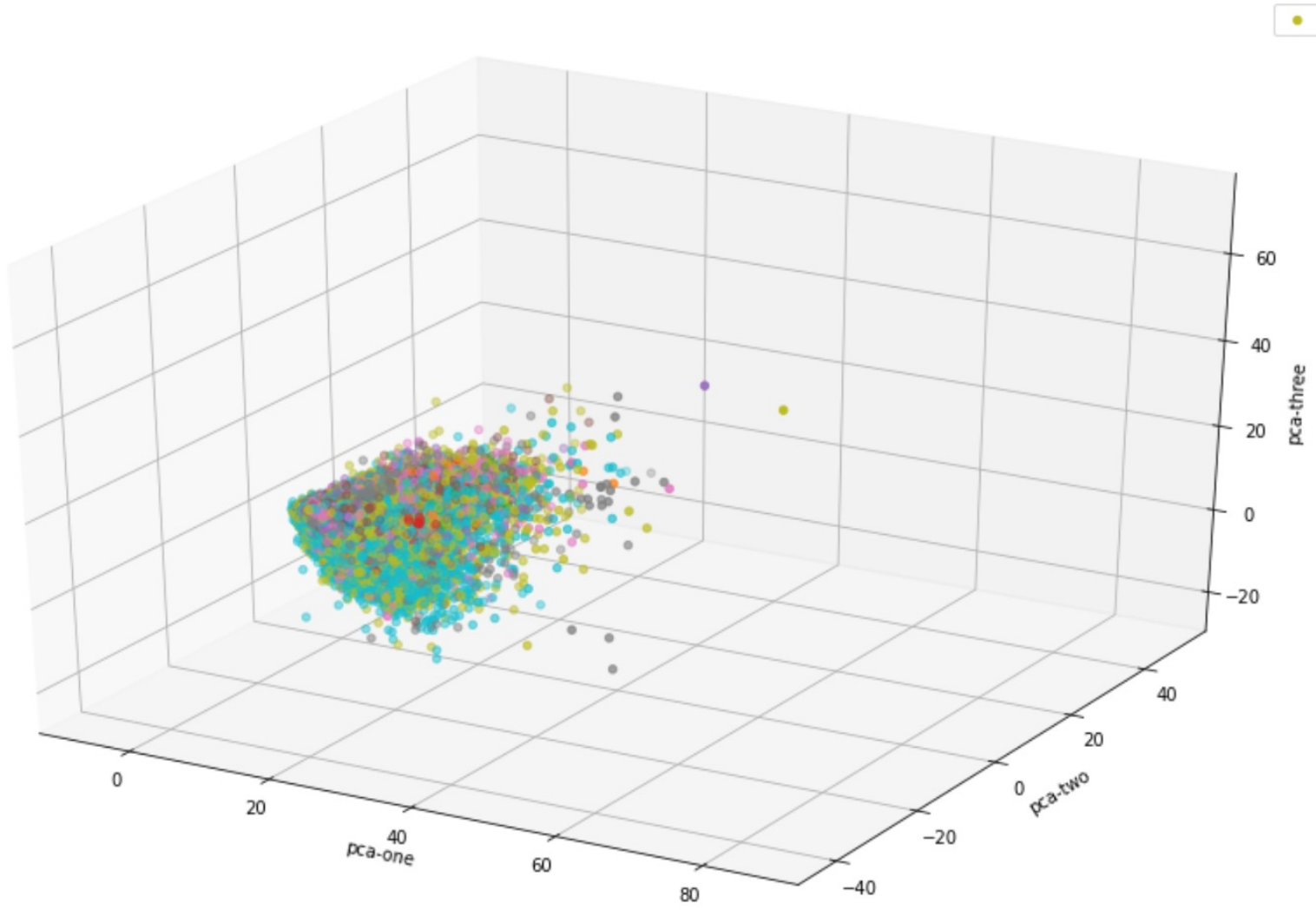
PCA

- For the PCA, we chose 20 components that gave us an explained variance of 60% which is the minimum required to work with a dataset according to internet. Indeed, the less variable we have, the faster our model will work.

The pca allowed us to reduce our number of variable from 90 to 20 which was very usefull considered the number of instances we have.



Here is a scatterplot of our data set in the 3 first components dimensions.



Models

- We are now ready to build our models. We decided to do :

- RandomForestClassifier
- kNN
- SVC
- NeuralNetwork

To have better models, we used GridSearchCV which allows us to tune our models to have the best parameters

RandomForestClassifier

- This is the first model that we tried. We didn’t succeed to apply GridSearchCV on it. Thus we tested every parameters manually.
- Here we have an example with a low-instance training and testing set to improve the speed .

- Our best score is 0.585 with a target of decade.

```
depths = [5,10,15,20]
estimators=[10,20,100]

for d in depths:
  for n in estimators:
    RFC = RandomForestClassifier(n_estimators=n, max_depth=d, random_state
    pred_y = RFC.predict(X_test)
    score = RFC.score(X_test, y_test)
    print("n_estimators:{0:.0f}, Depth:{1:.0f}, Score:{2:.3f}"
        .format(n, d, score))
```

```
n_estimators:10, Depth:5, Score:0.584
n_estimators:20, Depth:5, Score:0.585
n_estimators:100, Depth:5, Score:0.585
n_estimators:10, Depth:10, Score:0.579
n_estimators:20, Depth:10, Score:0.582
n_estimators:100, Depth:10, Score:0.582
n_estimators:10, Depth:15, Score:0.560
n_estimators:20, Depth:15, Score:0.575
n_estimators:100, Depth:15, Score:0.586
n_estimators:10, Depth:20, Score:0.547
n_estimators:20, Depth:20, Score:0.565
n_estimators:100, Depth:20, Score:0.582
```

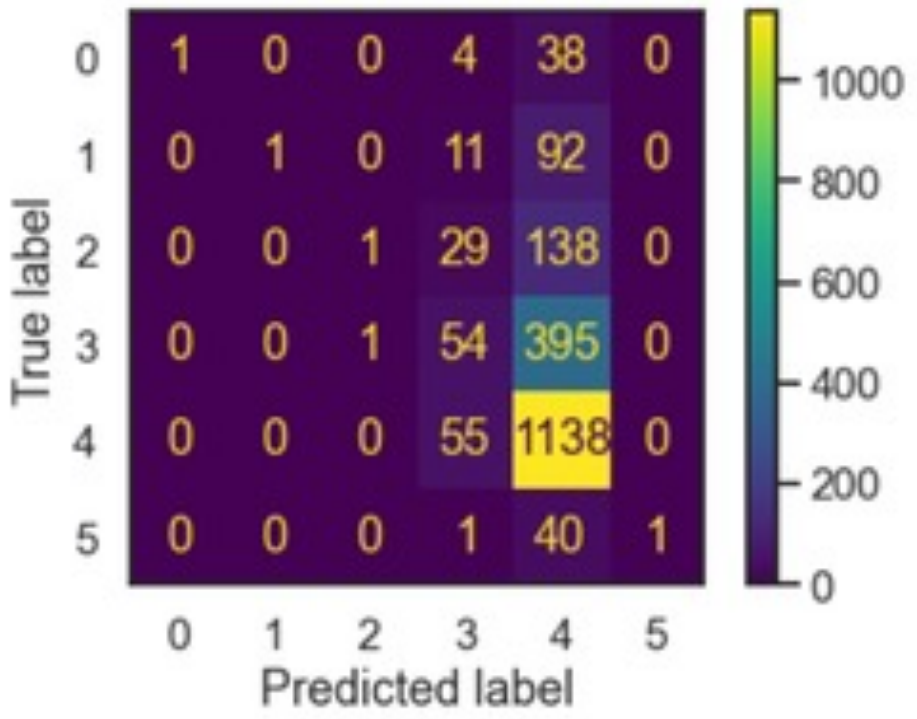
kNN

kNN is the 2nd model we built.
He was more effective than RandomForestClassifier.

Here is the confusion matrix of our kNN.

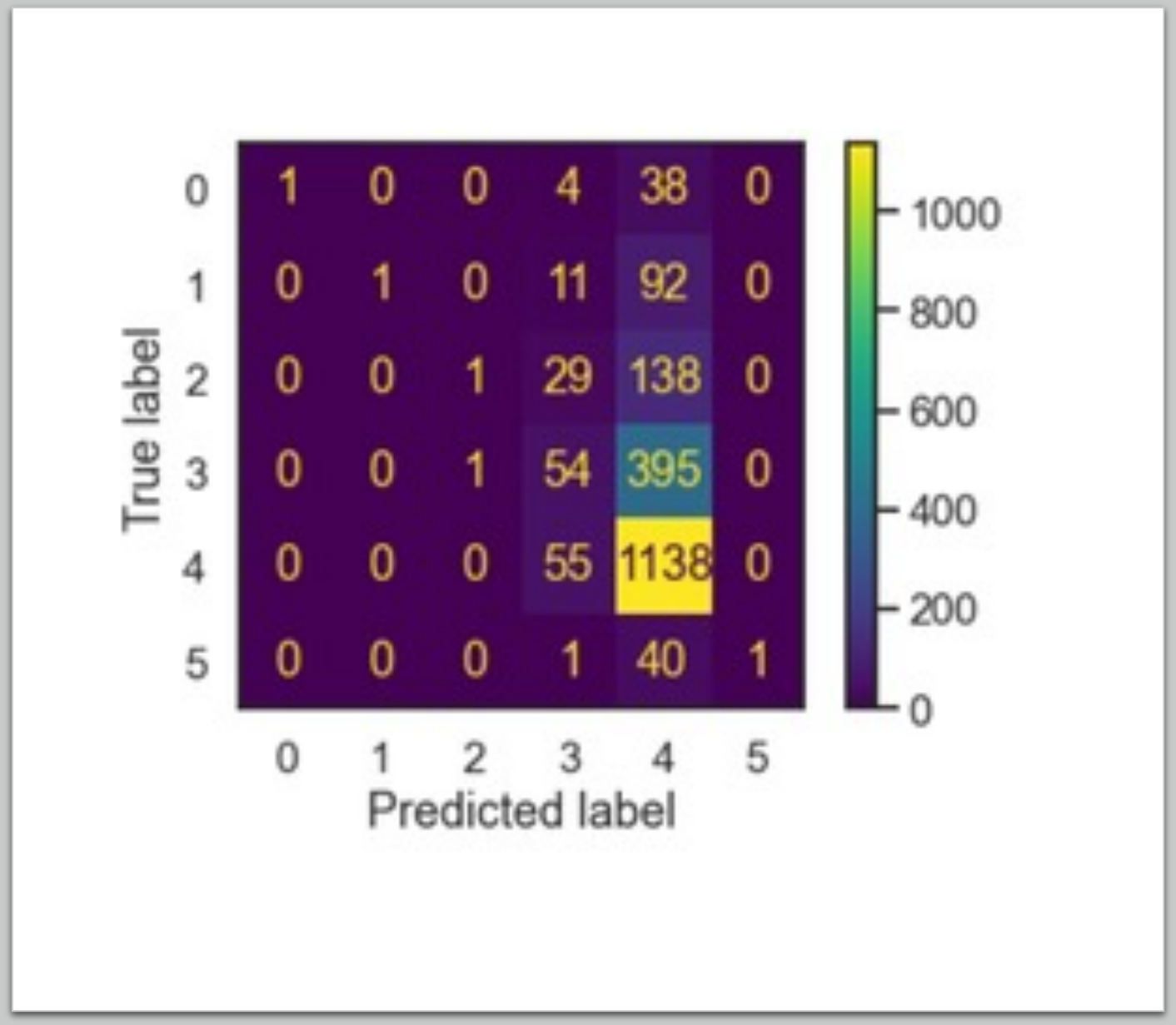
We chose the parameters k = 20 thanks to GridSearchCV.

The score is 0.598.



SVC

- Svc was the model that took the most time to compile. We decided to downsize our dataset after trying to get the predictions with it.
- Svc gave us a score of 0.5955
- Here is the correlation matrix



Neural Network

- The score of neural network is 0.58.
- This was the last model we tried, using deep learning.

	precision	recall	f1-score	support
1960	0.50	0.00	0.00	1082
1970	0.13	0.01	0.02	2389
1980	0.29	0.08	0.12	4018
1990	0.37	0.24	0.29	12512
2000	0.63	0.89	0.74	29086
2010	0.00	0.00	0.00	913
accuracy			0.58	50000
macro avg	0.32	0.20	0.20	50000
weighted avg	0.50	0.58	0.51	50000