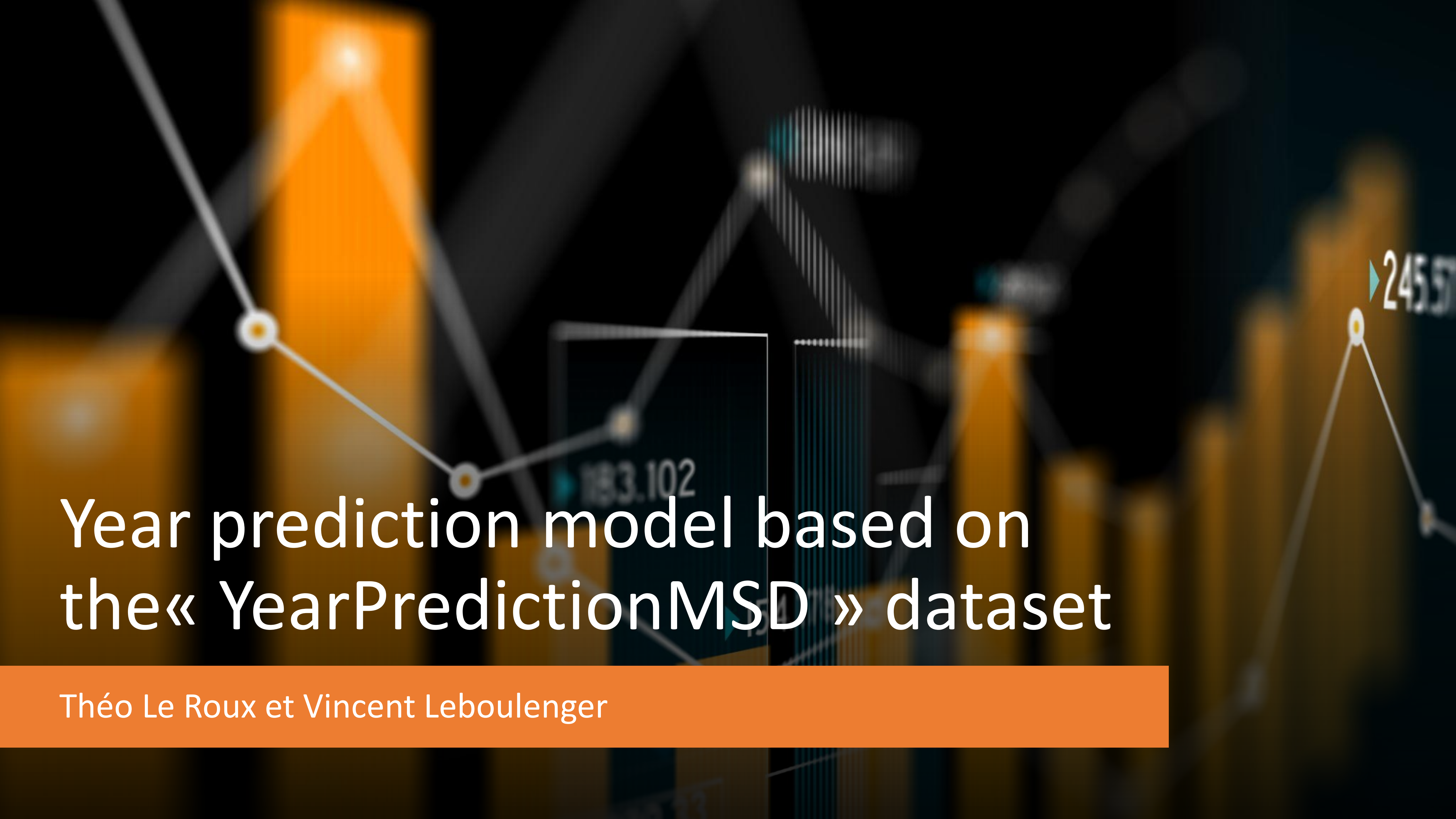
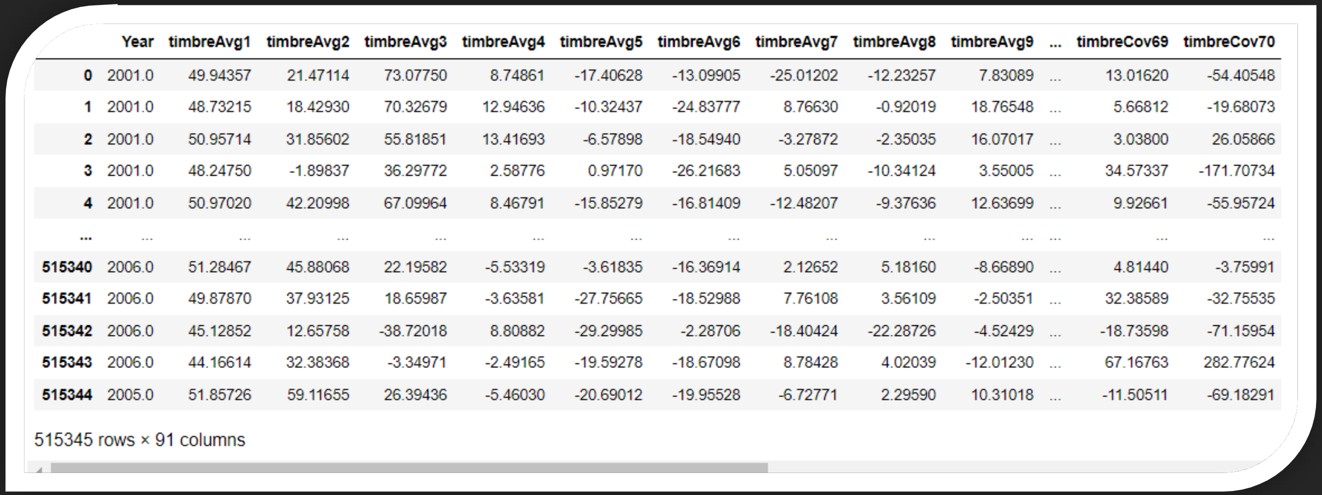


Year prediction model based on the« YearPredictionMSD » dataset

Théo Le Roux et Vincent Leboulenger

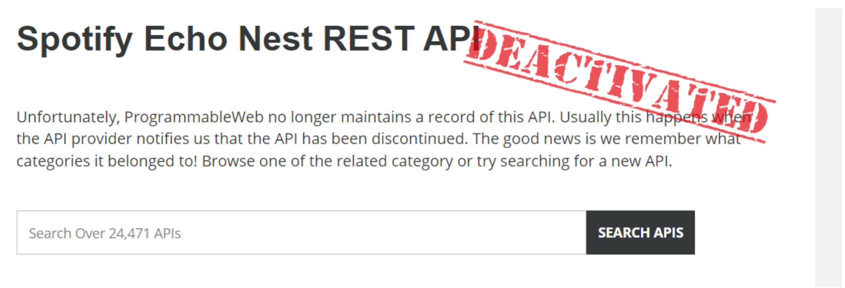
Dataset description

- => Timber features
- => Not accessible
- => No missing values



	Year	timbreAvg1	timbreAvg2	timbreAvg3	timbreAvg4	timbreAvg5	timbreAvg6	timbreAvg7	timbreAvg8	timbreAvg9	...	timbreCov69	timbreCov70
0	2001.0	49.94357	21.47114	73.07750	8.74861	-17.40628	-13.09905	-25.01202	-12.23257	7.83089	...	13.01620	-54.40548
1	2001.0	48.73215	18.42930	70.32679	12.94636	-10.32437	-24.83777	8.76630	-0.92019	18.76548	...	5.66812	-19.68073
2	2001.0	50.95714	31.85602	55.81851	13.41693	-6.57898	-18.54940	-3.27872	-2.35035	16.07017	...	3.03800	26.05866
3	2001.0	48.24750	-1.89837	36.29772	2.58776	0.97170	-26.21683	5.05097	-10.34124	3.55005	...	34.57337	-171.70734
4	2001.0	50.97020	42.20998	67.09964	8.46791	-15.85279	-16.81409	-12.48207	-9.37636	12.63699	...	9.92661	-55.95724
...	...	...	...	...	...	...	...	...	...	...	...	...	...
515340	2006.0	51.28467	45.88068	22.19582	-5.53319	-3.61835	-16.36914	2.12652	5.18160	-8.66890	...	4.81440	-3.75991
515341	2006.0	49.87870	37.93125	18.65987	-3.63581	-27.75665	-18.52988	7.76108	3.56109	-2.50351	...	32.38589	-32.75535
515342	2006.0	45.12852	12.65758	-38.72018	8.80882	-29.29985	-2.28706	-18.40424	-22.28726	-4.52429	...	-18.73598	-71.15954
515343	2006.0	44.16614	32.38368	-3.34971	-2.49165	-19.59278	-18.67098	8.78428	4.02039	-12.01230	...	67.16763	282.77624
515344	2005.0	51.85726	59.11655	26.39436	-5.46030	-20.69012	-19.95528	-6.72771	2.29590	10.31018	...	-11.50511	-69.18291

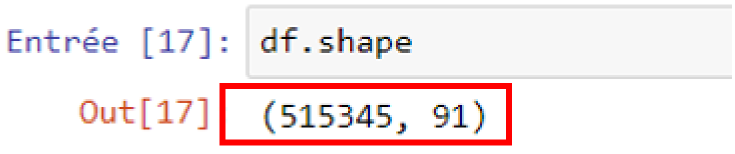
515345 rows × 91 columns



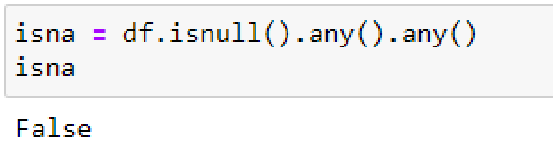
Spotify Echo Nest REST API

DEACTIVATED

Unfortunately, ProgrammableWeb no longer maintains a record of this API. Usually this happens when the API provider notifies us that the API has been discontinued. The good news is we remember what categories it belonged to! Browse one of the related category or try searching for a new API.



```
Entrée [17]: df.shape
Out[17]  (515345, 91)
```



```
isna = df.isnull().any().any()
isna

False
```

Dataset description

- Dataset content
- Model purpose

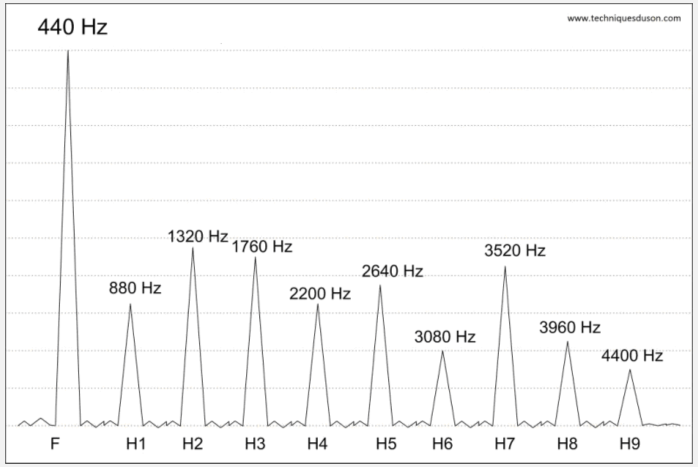
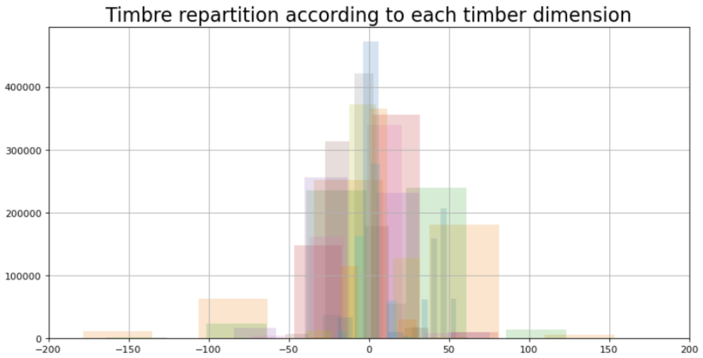


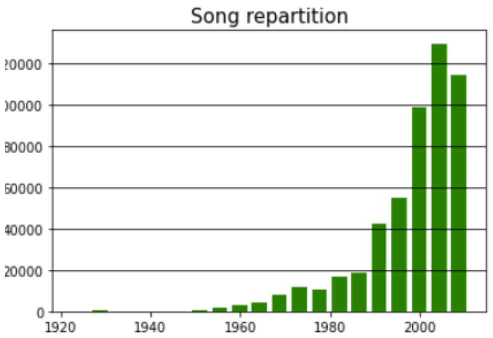
Illustration of music components

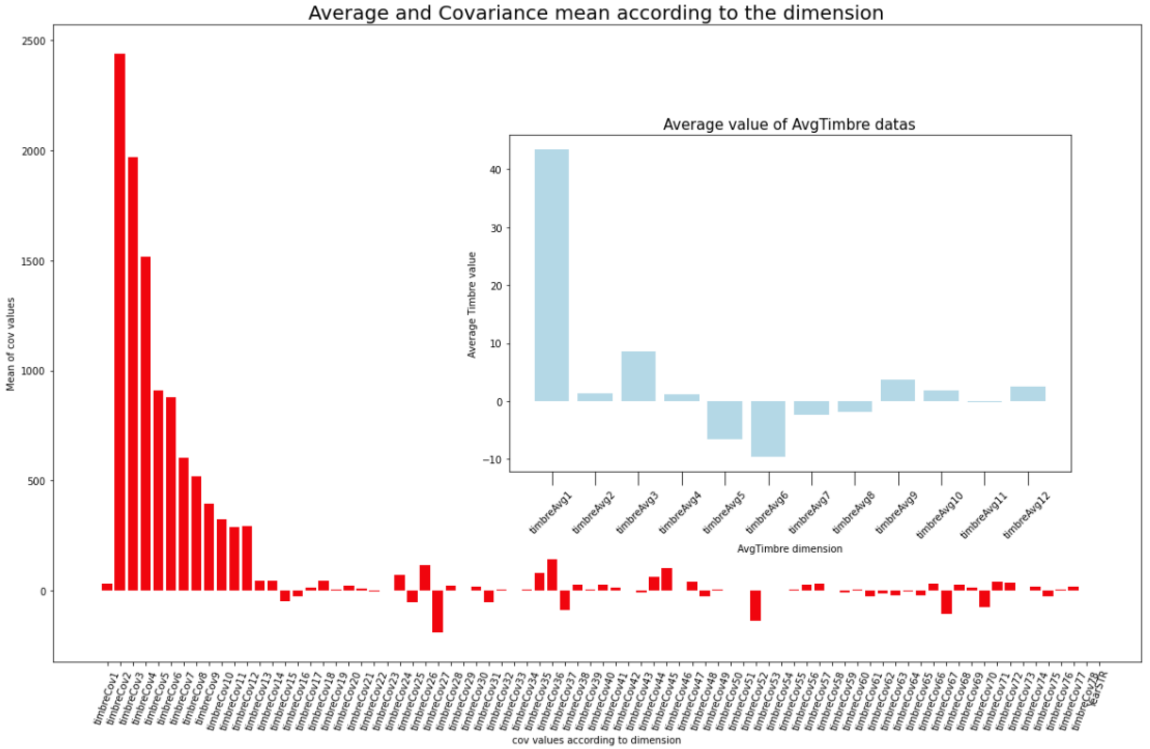
Illustration of timber

Data repartition

- Song repartition => biaised datas ?
- Values repartition => Normal distribution
- Some extreme columns => Normalization necessity

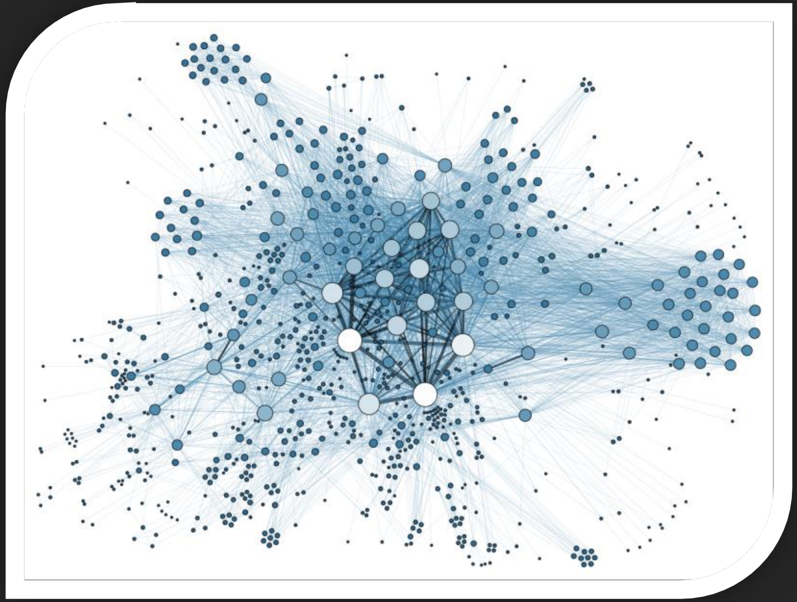

Timbre repartition according to each timber dimension

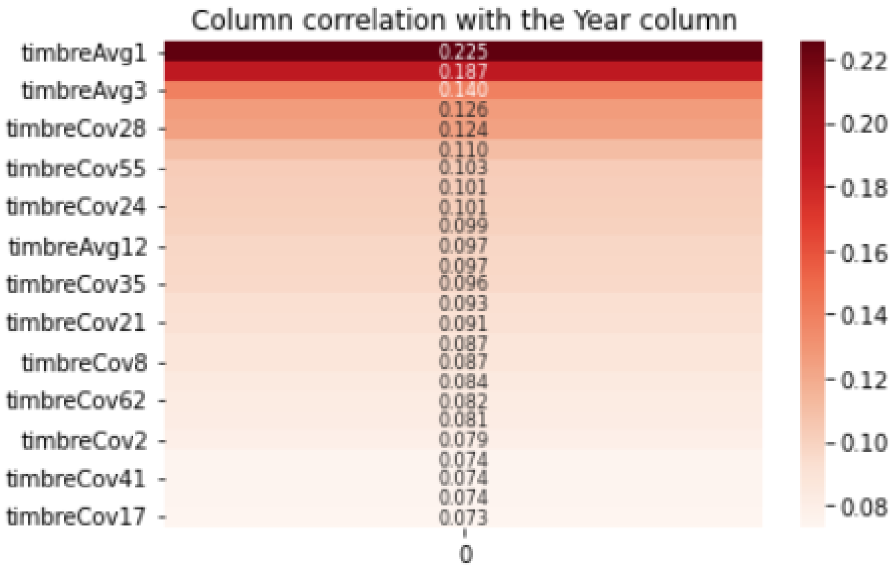

Song repartition

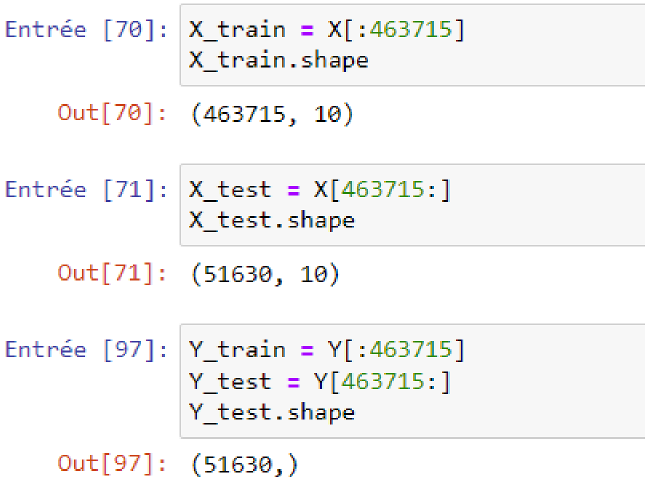

Average and Covariance mean according to the dimension

Data preprocessing

- No variable creation => not enough knowledge about our data
- Variable selection => compute correlation with year column
- Recommended website separation.





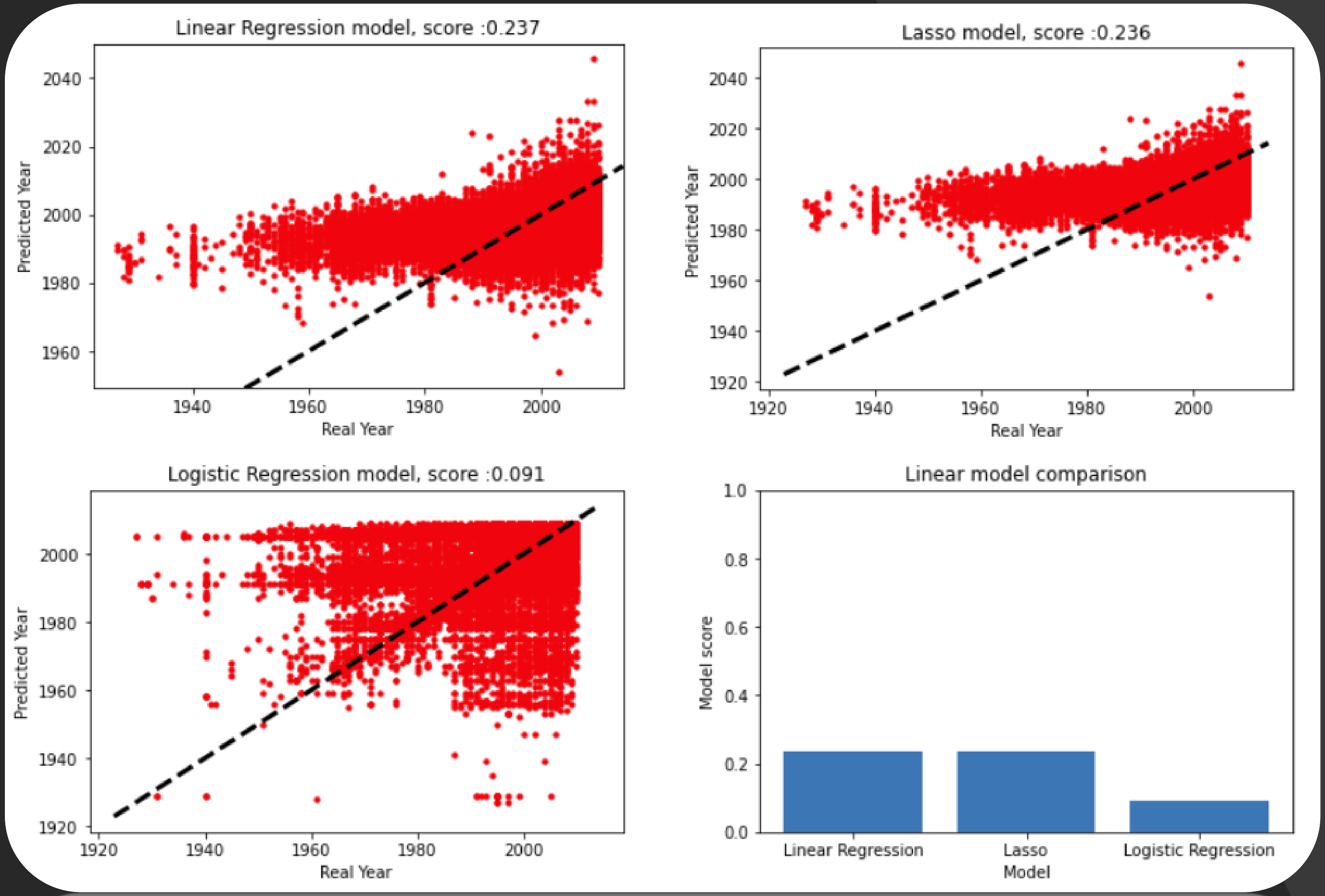


```
Entrée [70]: X_train = X[:463715]
             X_train.shape

    Out[70]: (463715, 10)

Entrée [71]: X_test = X[463715:]
             X_test.shape

    Out[71]: (51630, 10)

Entrée [97]: Y_train = Y[:463715]
             Y_test = Y[463715:]
             Y_test.shape

    Out[97]: (51630,)
```

Models creation : Linear models

- Linear model non-persuasive
- Trained with every features

(best scores)



Linear Regression model, score :0.237

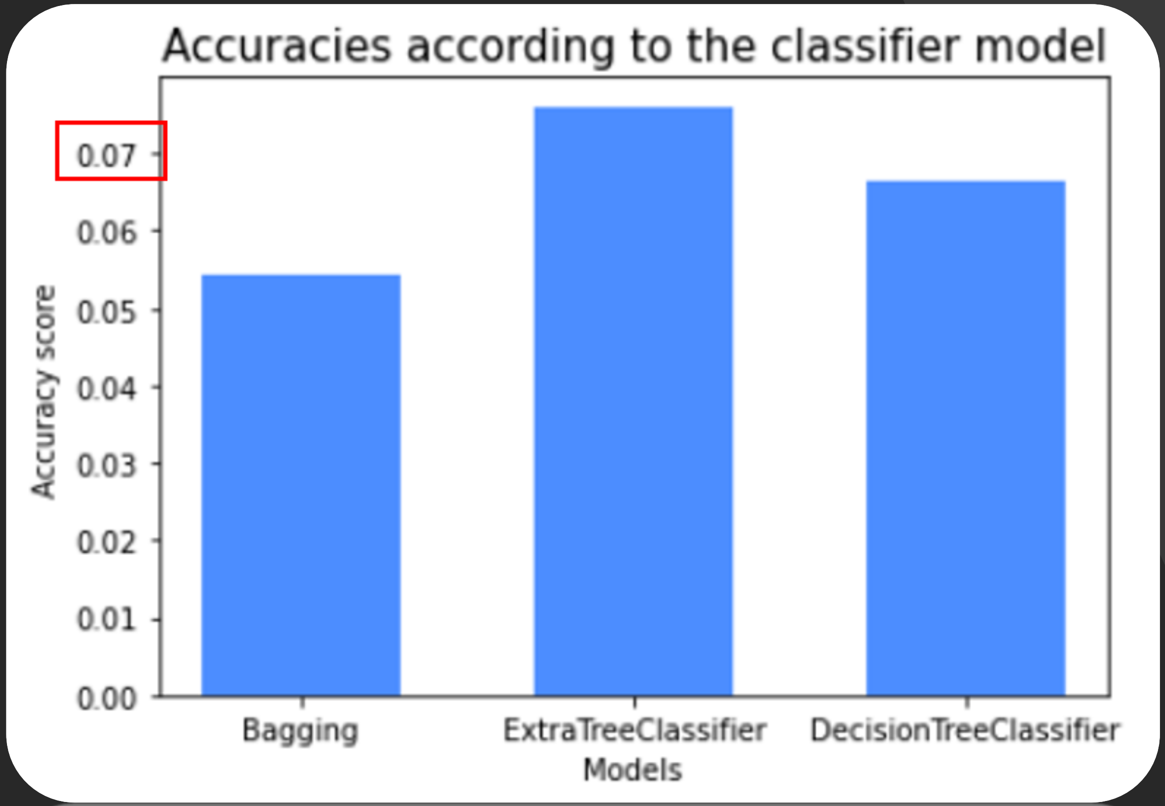
Lasso model, score :0.236

Logistic Regression model, score :0.091

Linear model comparison

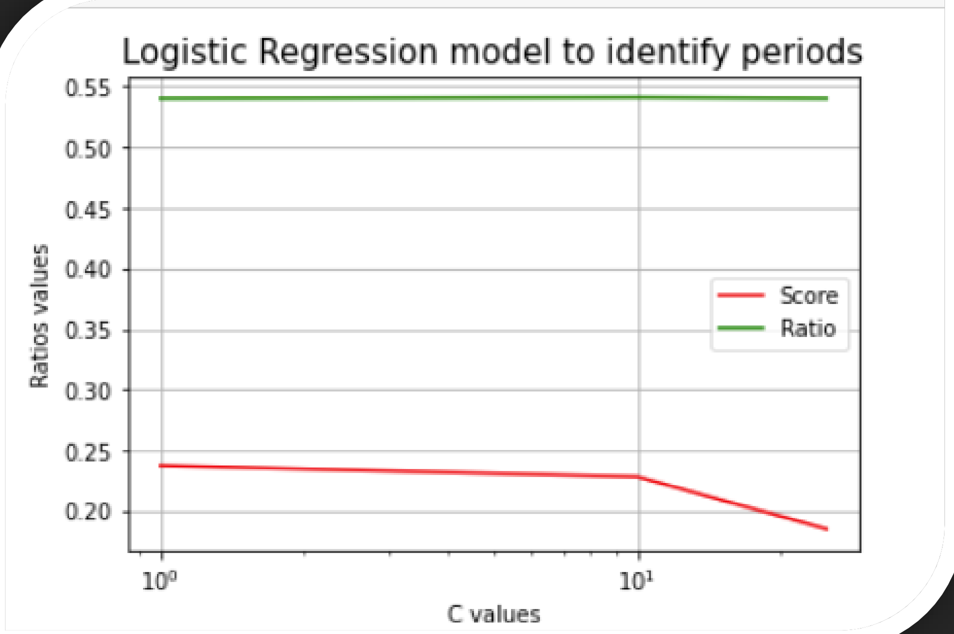
Models creation : Classifier

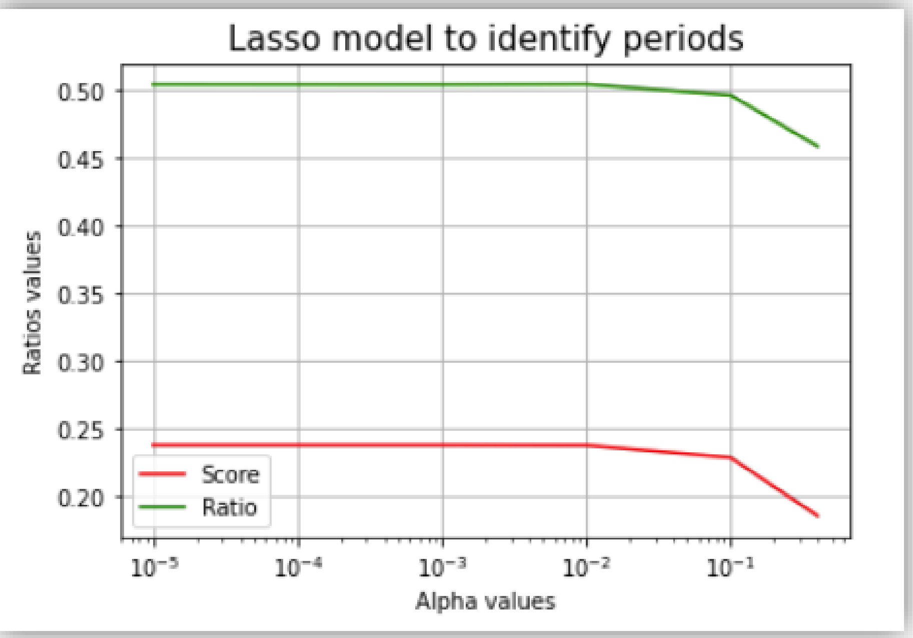
- Classifier model => Hard to be precise
- Trained with bests features

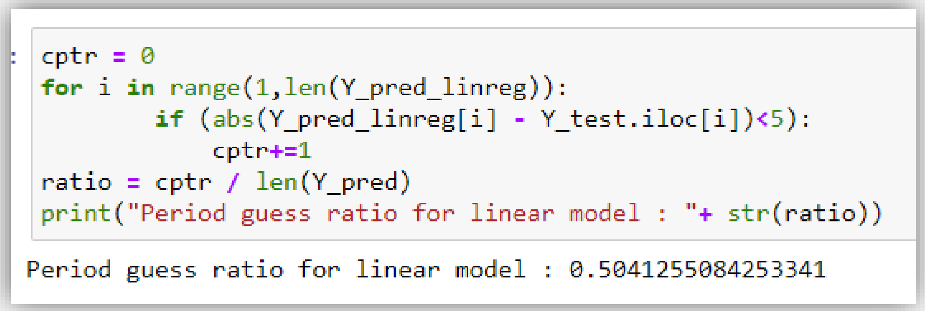


Focus on periods : Linear models

- => Model Logistic Regression with low alpha
- ~0.542





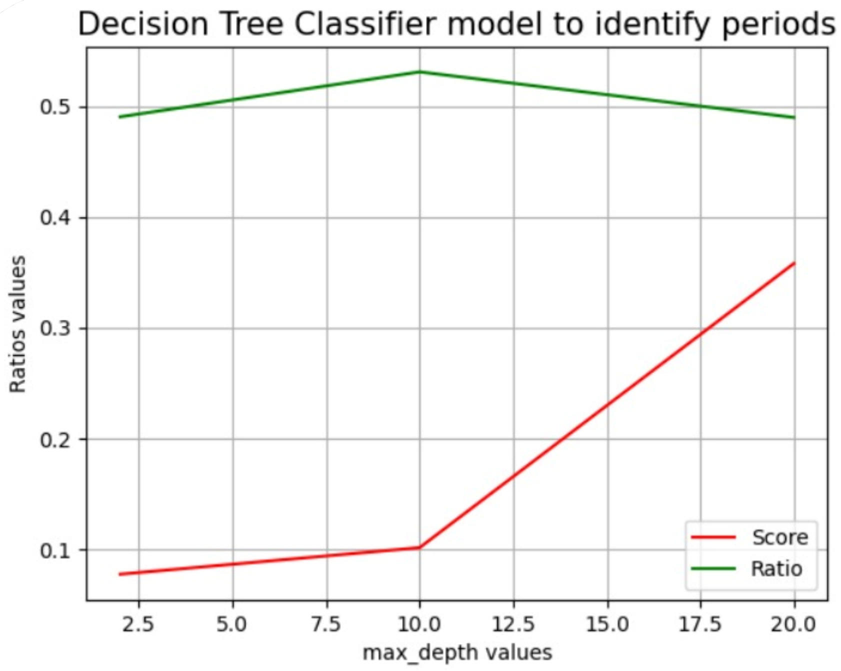


```
cptr = 0
for i in range(1,len(Y_pred_linreg)):
        if (abs(Y_pred_linreg[i] - Y_test.iloc[i])<5):
            cptr+=1
ratio = cptr / len(Y_pred)
print("Period guess ratio for linear model : "+ str(ratio))
```

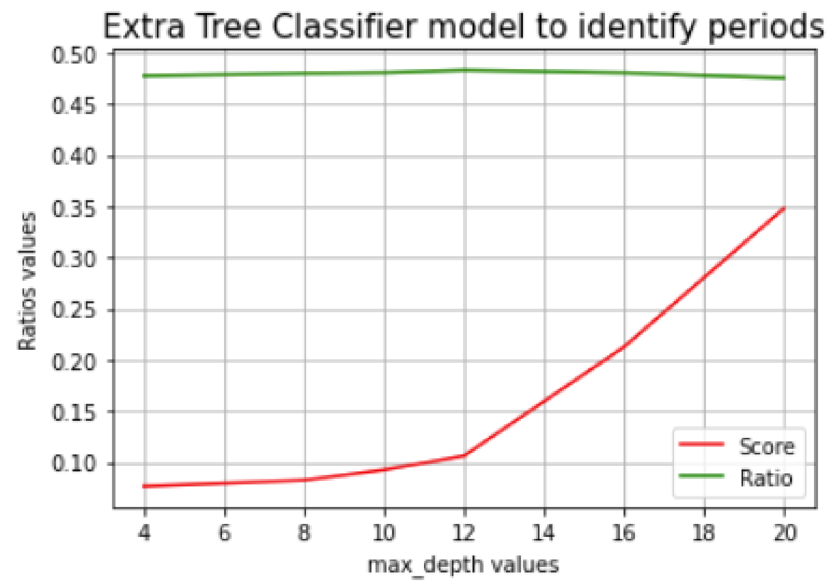
```
Period guess ratio for linear model : 0.5041255084253341
```

Focus on periods : Linear models

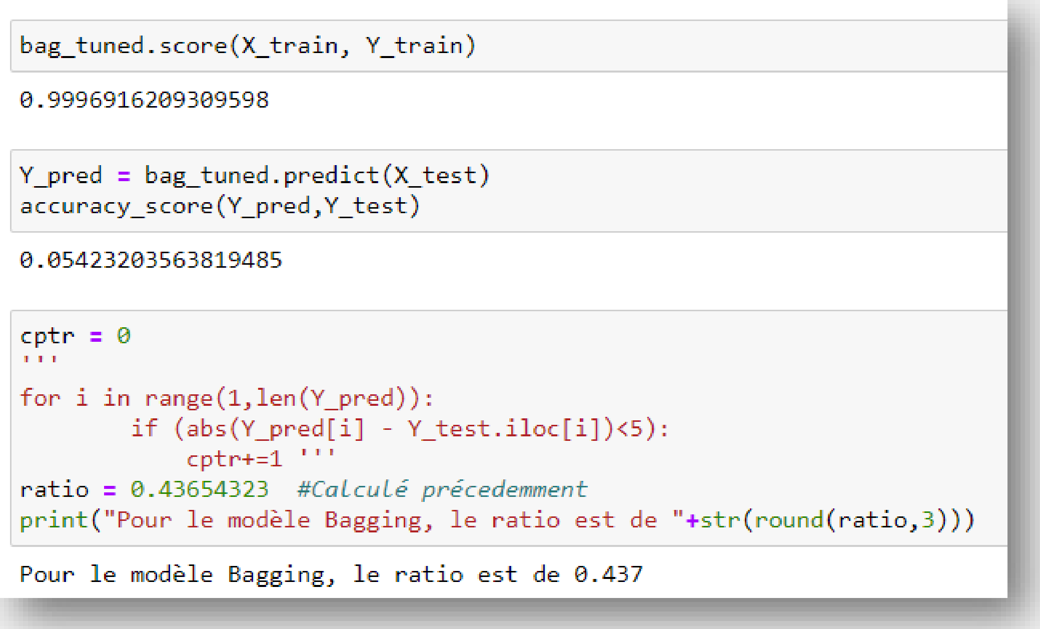
- => Model Decision Tree Classifier with a maximal depth of 10.
- ~0.531



```
Valeur max pour max_depth = 10 : 0.531
```



```
Valeur max pour max_depth = 12 : 0.483
```



```
bag_tuned.score(X_train, Y_train)
```

```
0.9996916209309598
```

```
Y_pred = bag_tuned.predict(X_test)
accuracy_score(Y_pred,Y_test)
```

```
0.05423203563819485
```

```
cptr = 0
'''
for i in range(1,len(Y_pred)):
        if (abs(Y_pred[i] - Y_test.iloc[i])<5):
            cptr+=1 '''
ratio = 0.43654323  #Calculé précedemment
print("Pour le modèle Bagging, le ratio est de "+str(round(ratio,3)))
```

```
Pour le modèle Bagging, le ratio est de 0.437
```

Conclusion

Best model : Logistic Regression ~ 54.2% accuracy