

# Classifieur de sons

IN104 2022

# Contact et déroulement du projet

Contact : [Gwendal.PRISER@ensta-paris.fr](mailto:Gwendal.PRISER@ensta-paris.fr) ou ENSTA : **R 2.20**

Repo GitHub contenant le projet : <https://github.com/gpensta/IN104>

## Outils nécessaires :

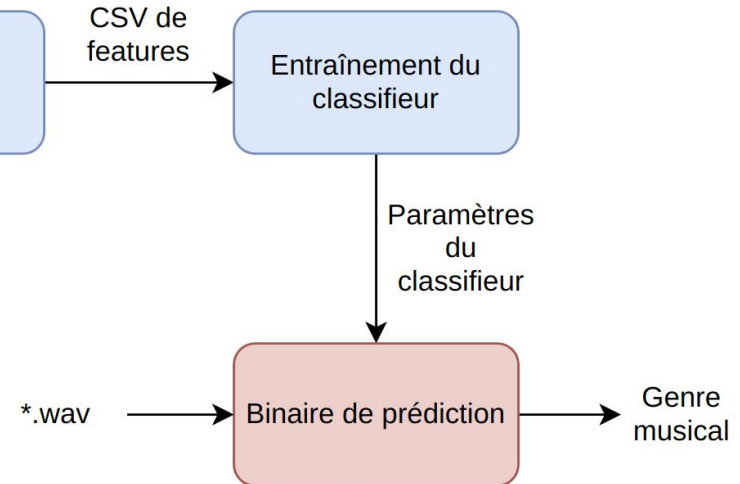
- **Linux** (Ubuntu, ...)
- Compilateur **GCC** (compilation du programme C)
- **Python 3** (entraînement d'un classifieur, outils de visualisation)
- Connexion internet

mardi 22/03	14:45 - 15:45		Cours magistral
mardi	16:00 - 18:00		TD info
mardi 29/03	14:45 - 18:00		TD info
mardi	16:00 - 18:00		TD info
mardi 05/04	14:45 - 16:45		cours magistral
mardi 12/04	14:45-16:45		TD info
mardi 19/04	14:45-16:45		TD info
mardi 3/05	14:45-16:45		TD info
mardi 10/05	14:45-16:45		TD info
mardi 24/05	14:45 - 18:00		Soutenance 2

# Projet

**Objectif** : Création d'un programme C classifiant des sons en genre musicaux.

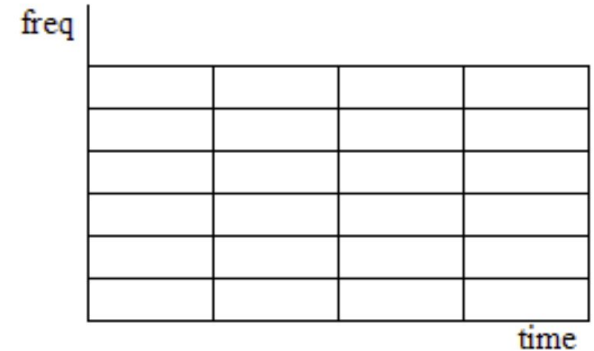
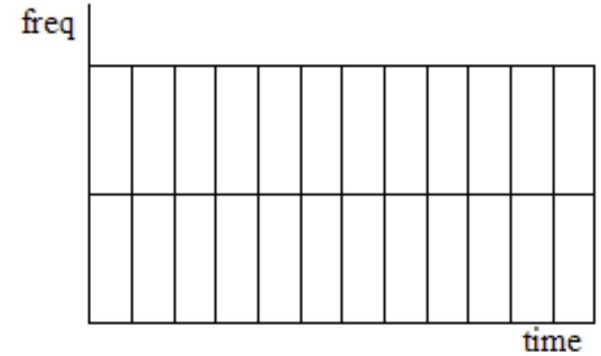
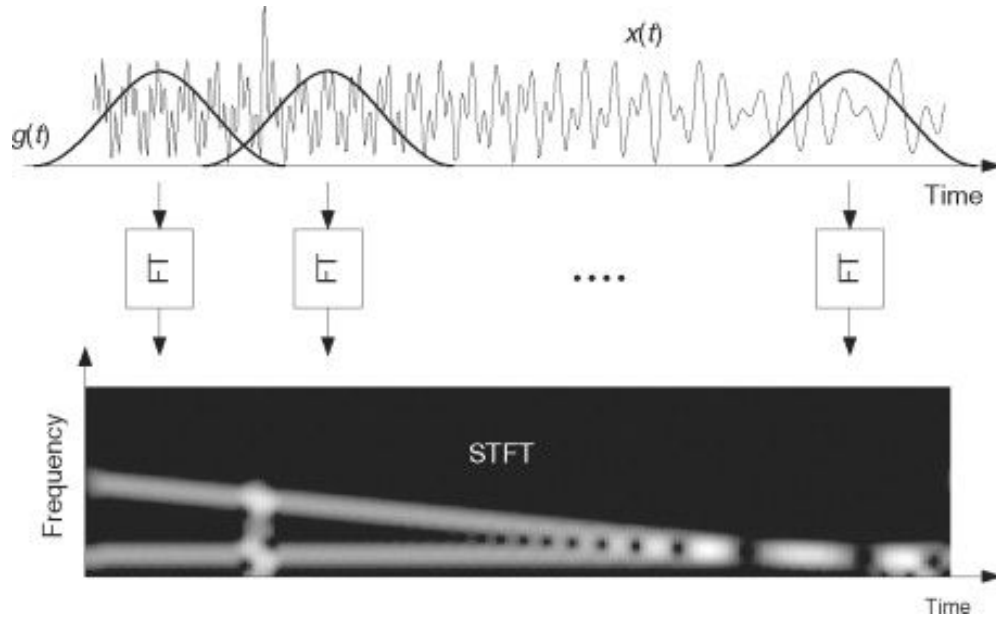
**Etapes du projet** :



(<http://marsyas.info/downloads/datasets.html>)

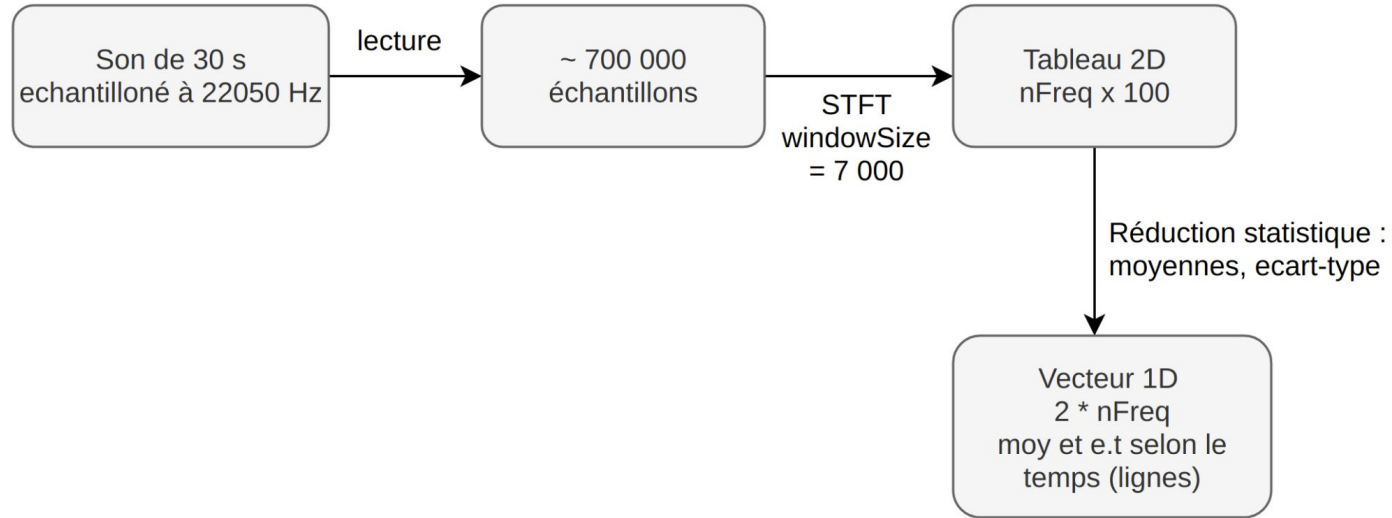
# Extraction des descripteurs (1)

## Short - Time - Fourier - Transform



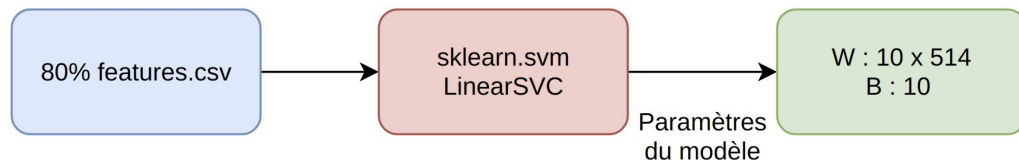
(<https://www.sciencedirect.com/topics/engineering/short-time-fourier-transform>)

# Extraction des descripteurs (2)



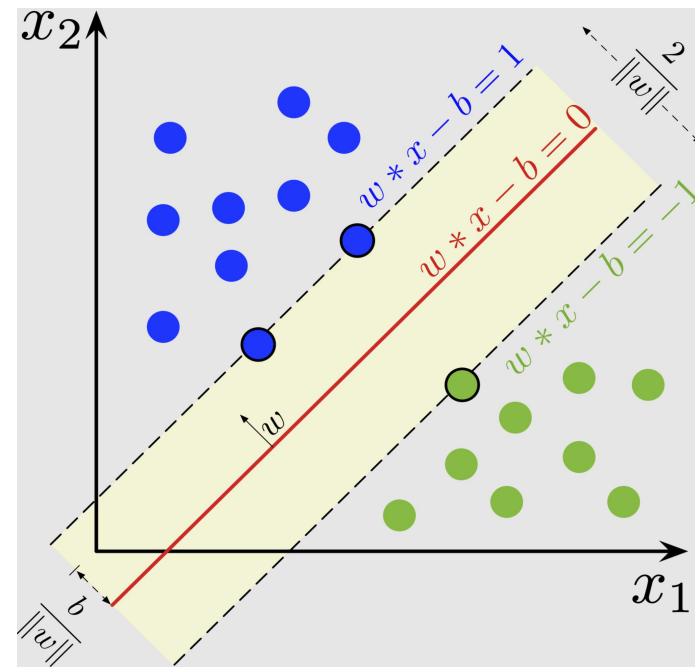
# Apprentissage supervisé et classifieur

$$\min_{\theta} \frac{1}{N} \sum_{i=0}^N (f_{\theta}(x_i) - y_i)^2$$

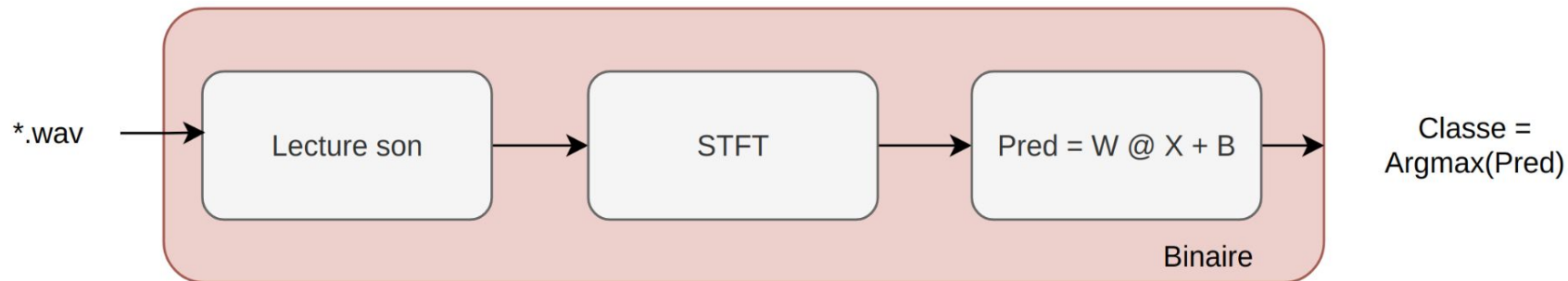


features.csv :

```
0; mu1; sigma1; ... ; mu257; sigma257;  
.  
.  
9; mu1; sigma1; ... ; mu257; sigma257;  
  
(100 x 10 lignes, 514 + 1 colonnes)
```



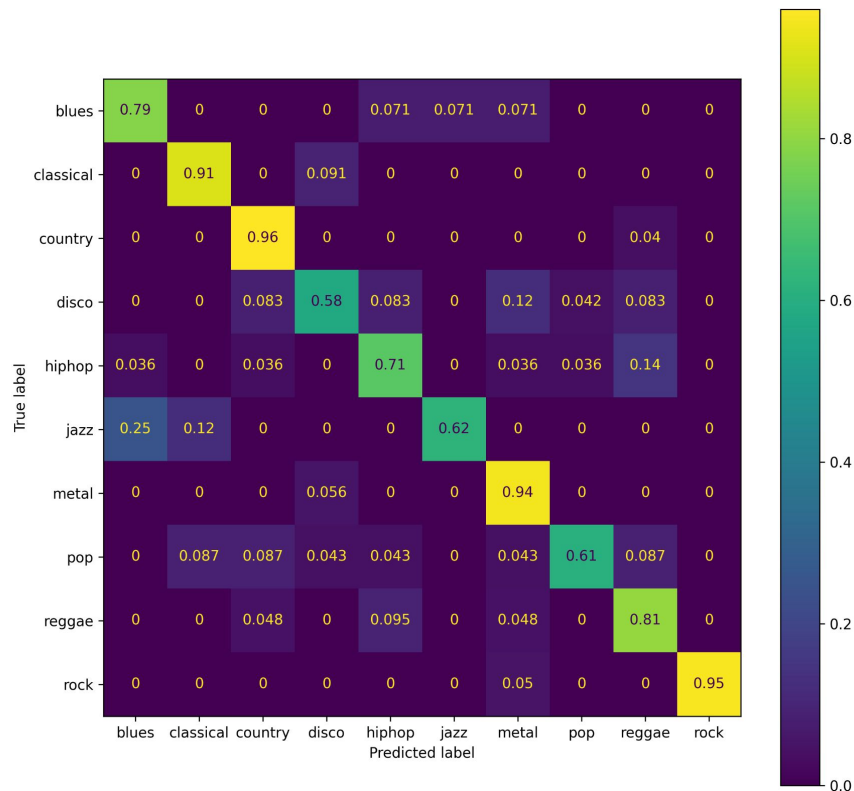
# Prédiction



```
~/IN104 ./classify song.wav
This song is classical music%
```

# Présentation des résultats

- Quelles sont les données utilisées ?
- Matrice de confusion
- Temps d'entraînement
- Temps d'inférence



(Exemple de scores obtenus avec un classifieur SVM linéaire)



# Architecture du projet

```
/IN104_Groupe
```

```
src/
```

```
main.c
```

```
include/ (modules du projet)
```

```
*.h
```

```
*.c
```

```
docs/
```

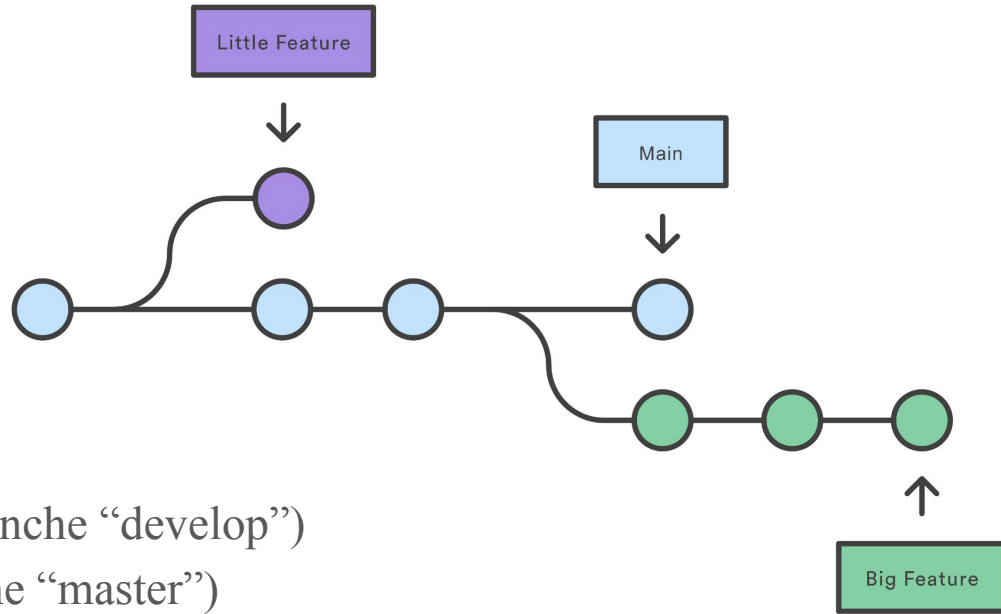
```
examples/
```

```
Makefile
```

```
README.md (Présentation du projet, instruction pour la compilation, comment  
exécuter le programme ?, résultats obtenus)
```

# Utilisation de Git

```
git init
git add <files>
git commit -m "<message descriptif>"
git remote add origin
git@github.com:<user>/<reponame>.git
git push origin master.
```



## Branching

```
git checkout -b develop (création de la branche “develop”)
```

```
git checkout master (se place sur la branche “master”)
```

```
git merge develop (fusion du dev effectué sur “develop” vers “master”, gérer avec minutie les potentiels conflits)
```

# Conseils

- Compiler son code et l'exécuter régulièrement.
- Valider les différents modules sur des exemples simples.
- Effectuer des commits réguliers.
- En groupe, effectuer une architecture logicielle en amont avec entrées / sorties des modules : respecter la convention de nommage. E.g. : `maVar`, `MaStruct`, `mon_module`, `CONSTANTE`.
- Utiliser un IDE (e.g. VSCode) pour pouvoir débbuger avec GDB.
- Effectuer ses recherches internet en anglais car plus de réponses sont disponibles.

# Suivi et évaluation du projet

## Tableau d'avancement :

[https://docs.google.com/spreadsheets/d/1iR\\_tJWAhS-ZisnpE-PrqFb7A\\_0W4x8w9xwW4t1x4P6A/edit?usp=sharing](https://docs.google.com/spreadsheets/d/1iR_tJWAhS-ZisnpE-PrqFb7A_0W4x8w9xwW4t1x4P6A/edit?usp=sharing)

## Soutenance

Date : Mardi 24 Mai de 14h45 à 18h

- Présentation des solutions techniques, des résultats ou pistes de recherche.
- Démonstration en direct : compilation du projet et inférence sur un exemple.

Durée maximale : 10 minutes, suivies de 4 minutes de questions.