

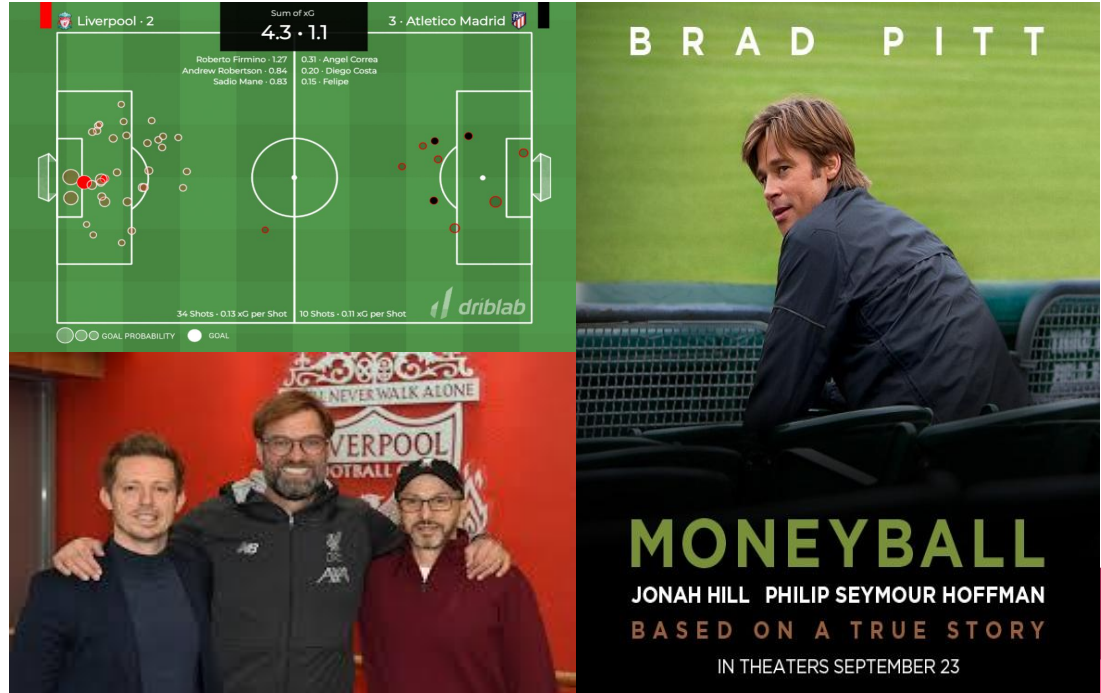
# Estimation de valeur de Footballeur



*In ballon veritas*

# Essor de la data dans le sport

- Les statistiques sont utilisées depuis un certain temps
- L'utilisation de la data a explosé ces dernières années
- Tout usage : analyse de match, de tactique, recrutement, formation...



# Nos objectifs

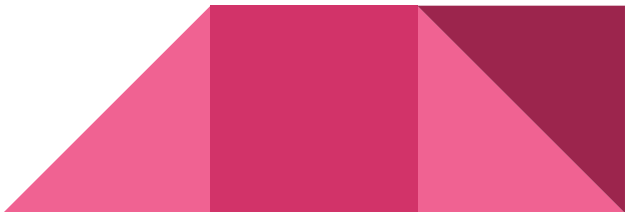
- Comprendre comment FIFA organise ses données
- Avoir une vue d'ensemble des différentes données
- Entraîner des modèles de prédiction des prix selon différentes méthodes
- Tenter d'estimer les prix de différents types de joueurs



# Dataset

Source :

[https://www.kaggle.com/stefanoleone992/fifa-21-complete-player-dataset?select=players\\_21.csv](https://www.kaggle.com/stefanoleone992/fifa-21-complete-player-dataset?select=players_21.csv)

- 7 fichiers au format csv
  - Données de joueurs de FIFA 15 à 21
  - Environ 15 à 20000 joueurs par jeu
  - 106 informations par joueur
- 

# Dataset

- 2 158023,<https://sofifa.com/player/158023/lionel-messi/210002>,L. Messi,Lionel Andrés Messi Cuccittini,33,1987-06-24,170,72,Argentina,FC Barcelona,Spain Primera Division,1,93,93,67500000,560000,"RW, ST, CF",Left,5,4,4,Medium/Low,Messi,Yes,138400000,"#Dribbler, #Distance Shooter, #FK Specialist, #Acrobat, #Clinical Finisher, #Complete Forward",CAM,10,,2004-07-01,2021,RW,10,85,92,91,95,38,65,,,,,,,,,"Finesse Shot, Long Shot Taker (AI), Speed Dribbler (AI), Playmaker (AI), Outside Foot Shot, One Club Player, Team Player, Chip Shot (AI)",85,95,70,91,88,96,93,94,91,96,91,80,91,94,95,86,68,72,69,94,44,40,93,95,75,96,,35,24,6,11,15,14,8,89+3,89+3,89+3,92+0,93+0,93+0,93+0,92+0,93+0,93+0,93+0,91+2,87+3,87+3,87+3,91+2,66+3,65+3,65+3,65+3,66+3,62+3,52+3,52+3,52+3,62+3
- 3 20801,<https://sofifa.com/player/20801/c-ronaldo-dos-santos-aveiro/210002>,Cristiano Ronaldo,Cristiano Ronaldo dos Santos Aveiro,35,1985-02-05,187,83,Portugal,Juventus,Italian Serie A,1,92,92,46000000,220000,"ST, LW",Right,5,4,5,High/Low,C. Ronaldo,Yes,75900000,"#Aerial Threat, #Dribbler, #Distance Shooter, #Acrobat, #Clinical Finisher, #Complete Forward",LS,7,,2018-07-10,2022,LS,7,89,93,81,89,35,77,,,,,,,,,"Power Free-Kick, Flair, Long Shot Taker (AI), Speed Dribbler (AI), Outside Foot Shot",84,95,90,82,86,88,81,76,77,92,87,91,87,95,71,94,95,84,78,93,63,29,95,82,84,95,,32,24,7,11,15,14,11,91+1,91+1,91+1,89+0,91+0,91+0,91+0,89+0,88+3,88+3,88+3,88+3,81+3,81+3,81+3,88+3,65+3,61+3,61+3,61+3,65+3,61+3,54+3,54+3,54+3,61+3

# Nettoyage, Parsing et Encoding

- Données très complètes, on enlève juste les joueurs sans contrat (0.3%)
- Le problème est bien plus la sélection de données
- A partir des années des fichiers d'origine, on modifie le nom des joueurs de "Gareth Bale" à "Gareth Bale 19" pour différencier les versions et calculer leur temps de contrat restant

```
for l in f :  
    num = re.search(r'\d+', fichier).group()  
    l = l.replace("\n", "")  
    l=l + "," + num + "," + str(int(num)+1999) + "\n"  
    fusion.write(l)
```

# Nettoyage, Parsing et Encoding

- On crée différents types de fichiers : un fichier fusionné (fonction fusion\_file) qui contient toutes les données et des fichiers spécifiques à une classe de joueur (only\_the\_chosen\_ones) qui contient une partie choisie, le train/test se fait aléatoirement (75/25) sur les différents fichiers.

**Parsing** : plus compliqué que ça en avait l'air

Tentative 1 :

```
return(list(func tools.reduce(lambda a,b: a+b, [[x] if i % 2 else x.split(",")  
for (i, x) in filter(lambda x: x, enumerate(line.split("\n")))], [] )))
```

Tentative 2 :

```
pattern = '(\\"[^\"]+\\"|([^\,]+)|(?=(,|,)))'  
temp = re.findall(pattern, line)  
res = ['' if s == "," else s for s in temp]
```



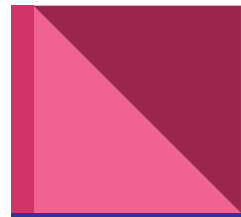
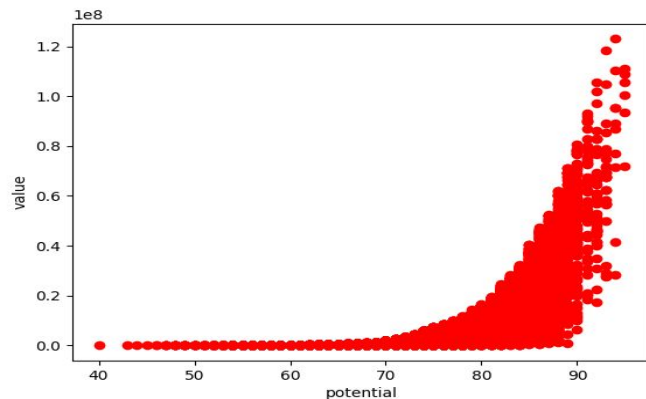
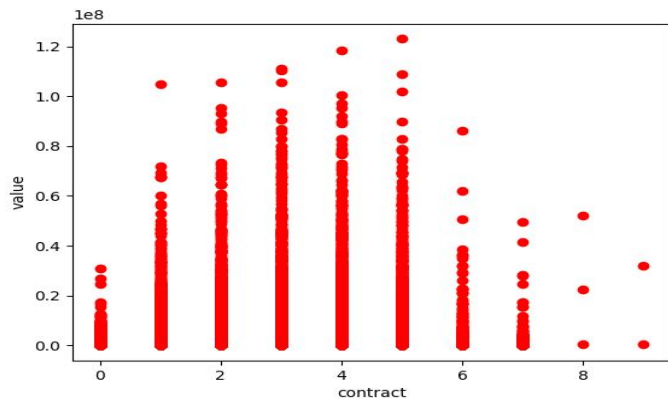
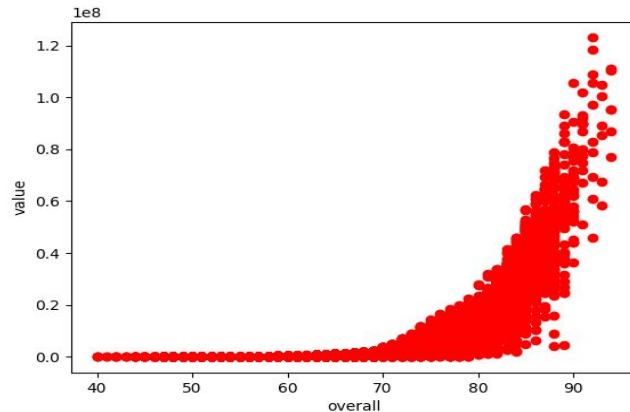
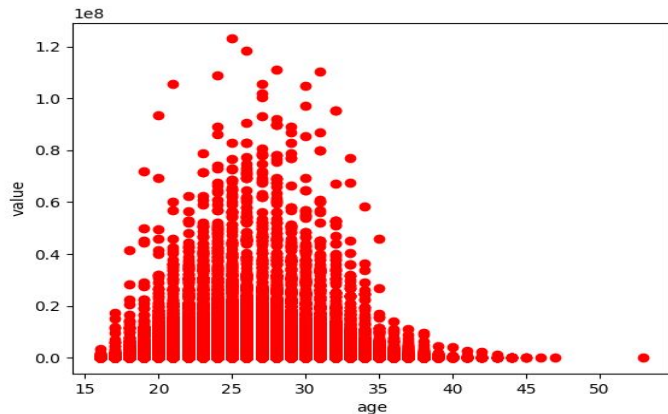
# Nettoyage, Parsing et Encoding

- On stocke les données de chaque joueur dans un grand dictionnaire dont la clef est le nom du joueur (et son année) et les valeurs une liste de nombreuses stats utiles ou moins utiles (longue liste commentée au début du code) avec les fonctions `parser_fifa` et `append_list_player`
- On pourra extraire nos données X et Y de ce dictionnaire avec `take_x_and_y`

```
l_numbers_to_append=[2,4,6,7,8,9,10,11,16,17,12,13,18,19,20,21,25,45]
for line in f:
    c+=1
    l=parse_line2(line)
    if l[30] != '': #On élimine les joueurs sans contrat (moins de 0,3%)
        list_player = append_list_player(l_numbers_to_append,l)
        dict_player[l[3] + " " + l[-2]]=list_player
    list_player = []
```



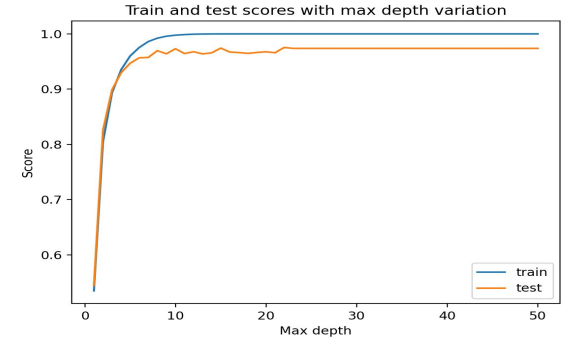
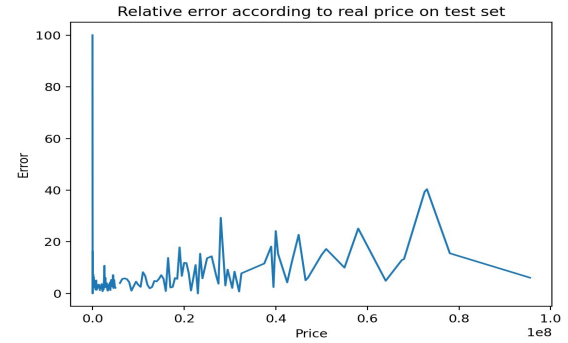
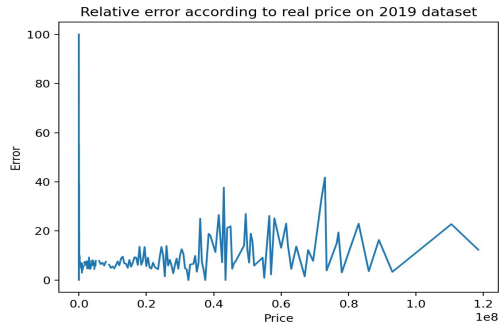
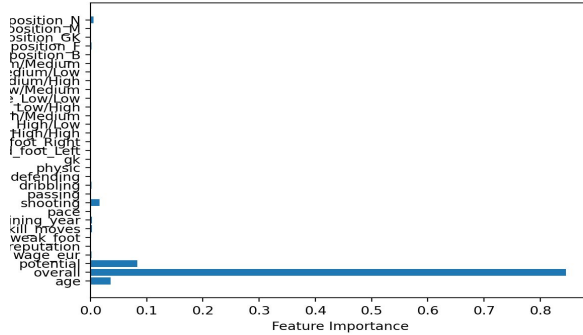
# On travaille surtout sur : âge, overall, potentiel, contrat



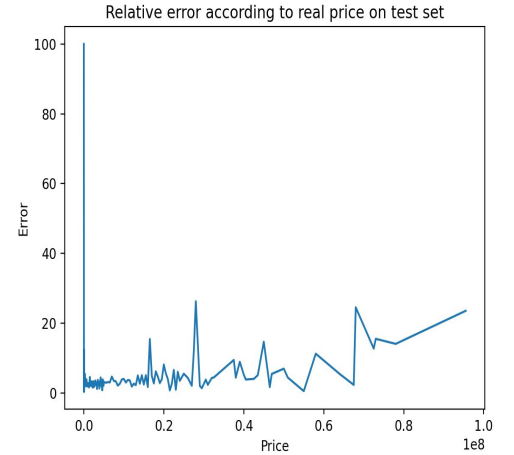
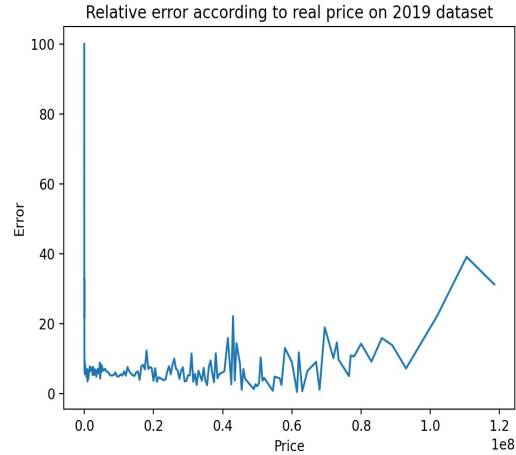
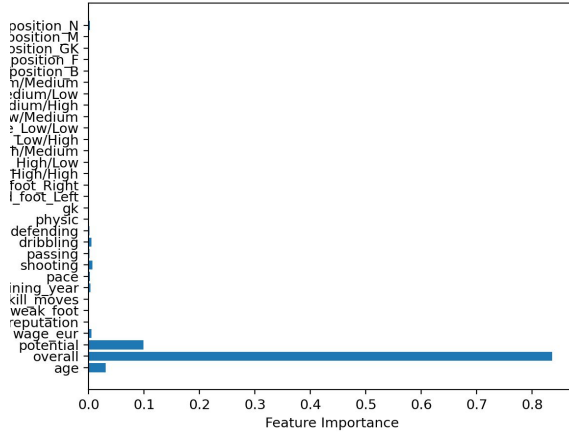
# Classification

- Dans un premier temps, on classe (classify\_value et éventuellement class\_to\_int) les joueurs en fonction de leurs valeurs et on essaie différentes méthodes de sklearn
- 5 différents types de Naive Bayes donnent des résultats moyens : précision de 0.8 pour 2 modèles, 0.3 pour les autres. Pour les plus grands joueurs, on tombe de 0.6 à 0. Les données sont trop interdépendantes.
- Une régression logistique donne des résultats similaires (score de 0.57) et rencontre un cap pour les plus grands (quasiment jamais au dessus de classe B+) malgré des résultats corrects pour les moins bons.
- On remarque plusieurs choses : le contrat est peu discriminant, la note overall est importante pour les joueurs les plus chers, tandis que ce sera plutôt l'âge et le potentiel pour les moins bons

# Decision Tree

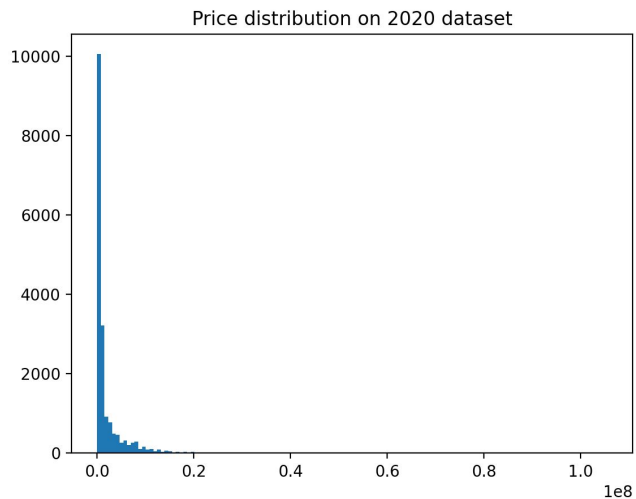


# Random Forest



# Conclusion distances

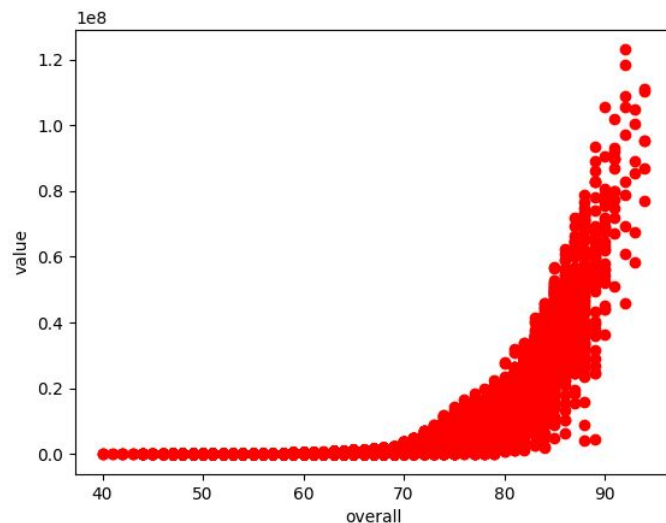
On obtient donc pour conclusion avec ces algorithmes de distance que plus le prix monte alors plus la prédiction est éloignée. Si le coût du joueur est élevé alors le modèle se trompera.



```
(venv) appleusers-MacBook-Air:Fifa_Project will$ python3 decision_tree.py
Training : 0.9975445260811825
Validation : 0.98845902785228
(venv) appleusers-MacBook-Air:Fifa_Project will$ python3 decision_tree.py
(venv) appleusers-MacBook-Air:Fifa_Project will$ python3 decision_tree.py
Training : 0.99996621951384
Validation : 0.9742060586089178
```

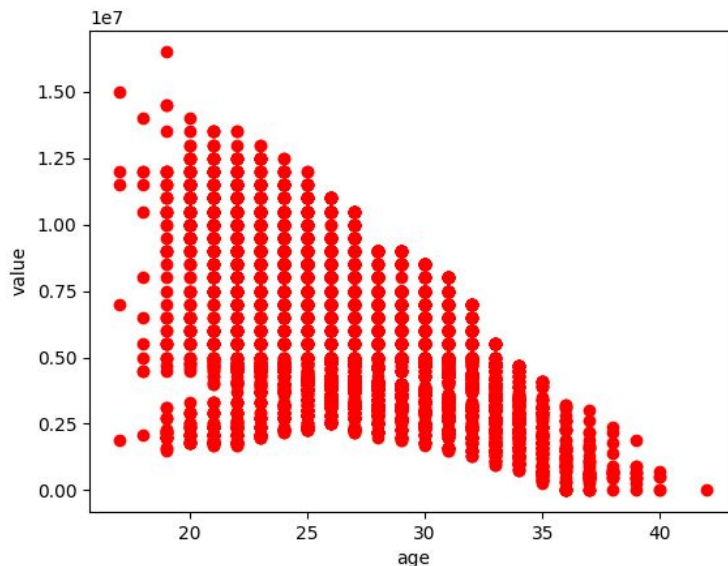
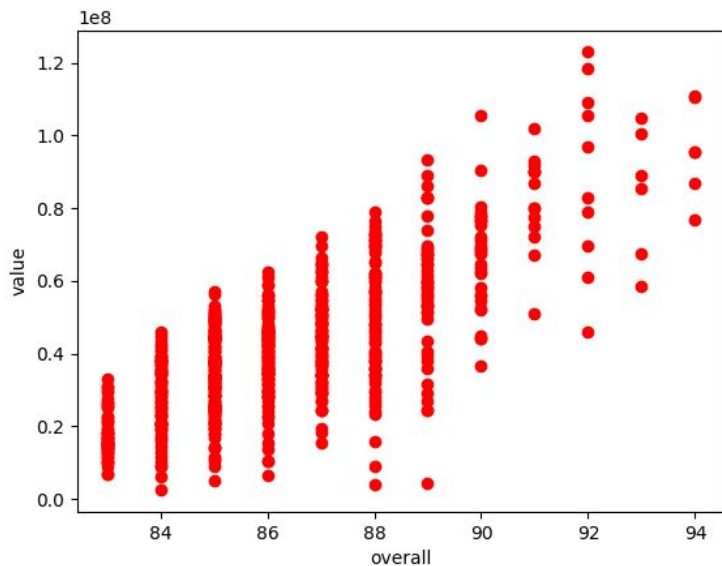
# Estimation de prix

- Comme attendu, la régression linéaire donne des résultats décevants, en particulier sur les meilleurs joueurs
- Elle rencontre un cap : aucun joueur ne dépasse les 15-20 M



# Estimation de prix

De bien meilleurs résultats en se concentrant et en entraînant sur un type de joueurs, y compris les meilleurs





# Recherche d'un outil plus global

- Hypothèse svr
- Difficulté à avoir un modèle cohérent
- Mauvaise convergence, le modèle rbf donnait de mauvais résultat
- Affaire à suivre ?



# On peut compter sur Neigh-mar

- Pour un résultat global, changement de paradigme
- Notre recherche marche mieux sur les éléments proches entre eux.
- On utilise KNearestNeighborsRegression, tout d'abord avec Minkowski et 5 voisins
- On obtient de très bons résultats (score > 0.90 et écart type moyen inférieur à 15%), que ce soit en général, en se restreignant à une classe ou à un seul jeu (cela ne vient donc pas de la proximité du joueur avec lui-même)
- Même Messi et Cristiano ronaldo donnent des résultats satisfaisants
- Seul Mbappe y échappe un peu : très jeune, haute note et haut potentiel

# Discussion

- Ce modèle peut être tuné avec des fonctions distance (comme `lin_dist` ou `overall_dist`) qui peuvent donner plus de poids à certaines statistiques et de la cross-validation
- Applicable pour la classification
- Prédire d'autres caractéristiques
- De même, on peut essayer avec données moins absolues comme celles dans la vraie vie : CIES, recrutement, analyse, etc..



# Contributions

- Idée du projet : commune (elle a pas mal changé), database Gaspar
- Générateur de fichiers : les deux (idées de séparation et implémentation)
- Parser ligne : (on s'est bien arraché les cheveux, mais c'est Gaspar qui a trouvé !)
- Parser fichier : les deux
- Analyse graphique et statistique : Gaspar
- Classifieur : Wilfried
- Modèles de classification : Wilfried
- Distances : Wilfried
- Modèles de régression : Gaspar
- Slides : Gaspar et Wilfried

