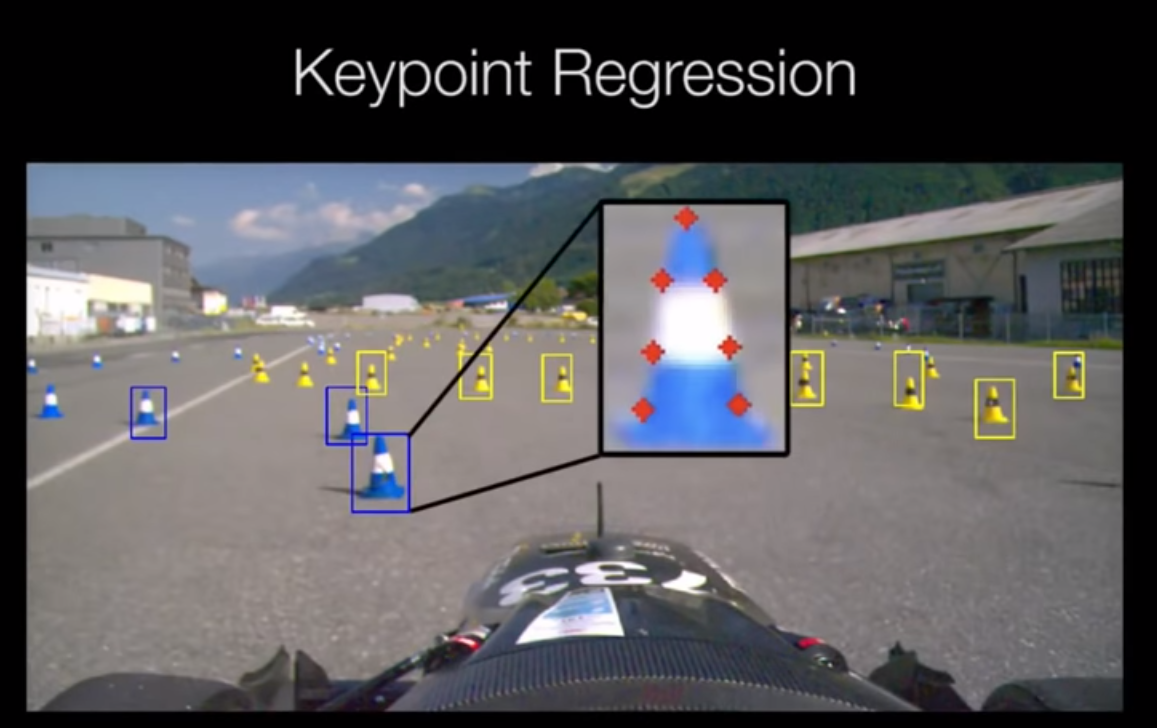
# ARECE 2 – Rapport Technique

1. Reconnaissance de l’environnement

* Contexte



La reconnaissance de l’environnement s’effectue avec une algorithme de keypoints qui va permettre de détecter les plots avec certitude. Cet algorithme place des « points clés » sur un objet pour permettre de le reconnaitre dans une scène aléatoire.

* Dans le code :

Cela correspond à la détection de feature du flux vidéo.

Une image contenant texte, capture d’écran, écran, argent

Description générée automatiquement

* Conclusion reconnaissance de l’environnement :

Nous sommes actuellement en train de travailler sur la détection de keypoints. C’est pourquoi la réunion du 20/06 nous permettra d’avancer sur la détection de keypoints à travers le flux vidéo et ainsi d’évaluer avec certitude la présence ou non d’un plot.

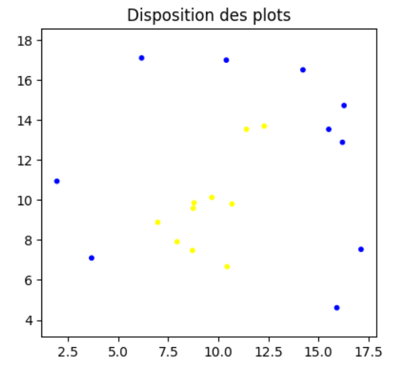
Avant de calculer la nouvelle trajectoire, nous insérons les positions des plots dans deux fichiers distincts : un pour les plots jaunes et un pour les plots bleus.

1. Calcul de la trajectoire optimale

Une fois les positions des plots récupérées et enregistrées, nous avons décidé de mettre en place un algorithme de prédiction SVM pour calculer une trajectoire optimale du véhicule.

Ce sont des algorithmes d’apprentissage utilisés en machine learning pour résoudre des problèmes de classification, de régression ou de détection d’anomalie.

Les conditions d’utilisation de ces algorithmes sont les suivantes :



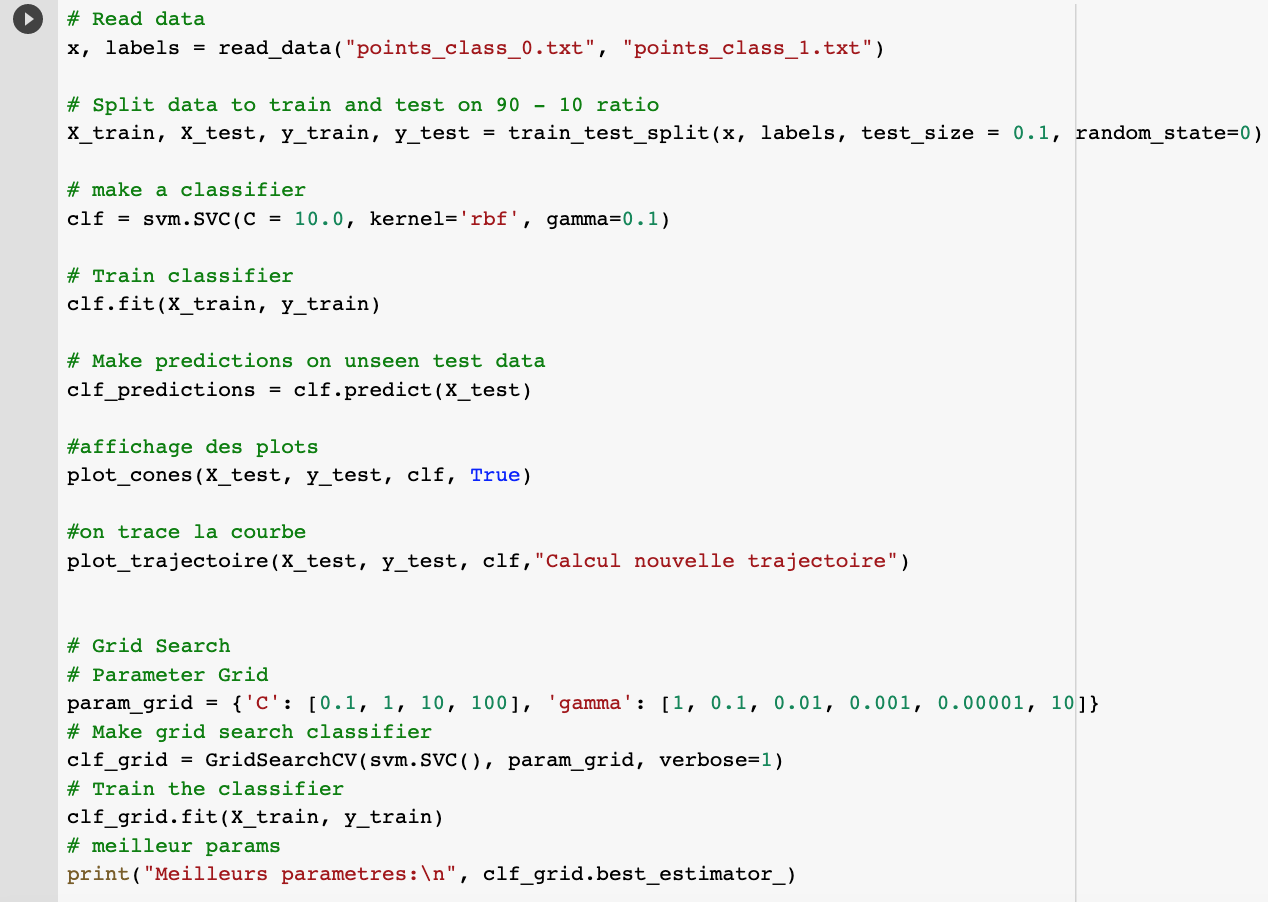
- D’avoir des classes distinctes : un fichier pour les plots bleus et un fichier pour les plots jaunes.

- Et une zone de frontière qui délimite ces classes (une zone qui sépare bien d’un côté les plots jaunes et de l’autre les plots bleus).

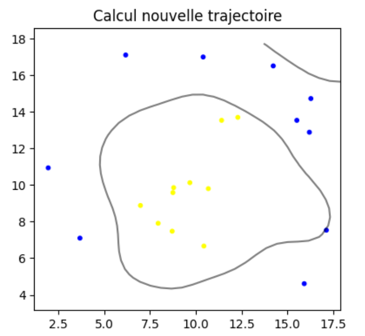
Nous entraînons le modèle avec des paramètres fixés.

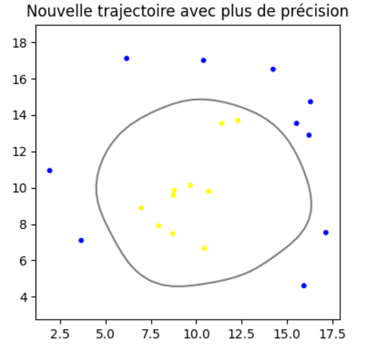
90% des données sont utilisés pour entrainer le modèle. Et 10% pour le tester afin de prédire la trajectoire.

Puis nous affichons le premier résultat. Afin de gagner en précision lors de chaque itération, nous pouvons jouer sur les hyper-paramètres « c » pour contrôler l'erreur et « gamma » pour la gestion de la courbure de la courbe.



Résultats :

1er résultat pour c=10 et gamma=0.1

2e résultat pour c=1 et gamma=0.1

Le code est disponible à l’adresse : <https://github.com/Paulcou/ARECE2/tree/main/code_calcul_nouvelle_trajectoire_SVM>

Commande pour actionner le projet :

Dans le repo : ARECE2\_suite

Commande : python3 exemple\_code.py

## Github : <https://github.com/Paulcou/ARECE2>/