Travail préparatoire TP

Les fonctions suivantes doivent avoir comme argument des listes.
ici L1 et L2.
Définir les fonctions suivantes.
#Faire les commentaires

```
def Variance(L1):
def Moyenne(L1):
def Covariance(L1, L2)
def EcartType(L1):
```

cf le cours
La fonction suivante Regression_lineaire(L1,L2) doit retourner alpha et beta.
Elle doit être coder à l'aide des fonctions précédentes.

```
 def Regression_lineaire(L1,L1) :
```

Implémentation de l'algorithme des k plus proches voisins

Partie 1

Calcul de la distance entre deux points

Ecrire une fonction distance prenant en paramètre deux points (un point sera représenté par un tuple (x,y)) et renvoyant la distance entre ces deux points.

$$\boldsymbol{distant(PointA; PointB)} = \sqrt{(\boldsymbol{x_a} - \boldsymbol{x_a})^2 + (\boldsymbol{y_a} - \boldsymbol{y_b})^2}$$

Calcul des distances de tous les points à une origine donnée

1) Construire une fonction **k_proches_voisins** qui prend en arguments :

- le jeu de données
- les coordonnées du point origine dont on cherche à déterminer la nature
- la valeur de k : nombre de plus proches voisins à considérer et renvoyant la liste des k plus proches voisins sous forme d'un tuple (distance, propriété)
- **def k_proches_voisins(data, origine,k):**

Prédiction de la valeur du caractère en fonction des k plus proches voisins

1) Déduire de la fonction précédente la fonction **categorie_devine** qui prend en arguments :

- le jeu de données
- les coordonnées du point origine dont on cherche à déterminer la nature
- la valeur de k : nombre de plus proches voisins à considérer et qui renvoie la propriété estimée par l'algorithme des k plus proches voisins, à savoir la propriété la plus fréquente parmi les k plus proches voisins.

En cas d'égalité, on choisira la couleur dont la somme des distances correspondante est la plus petite.

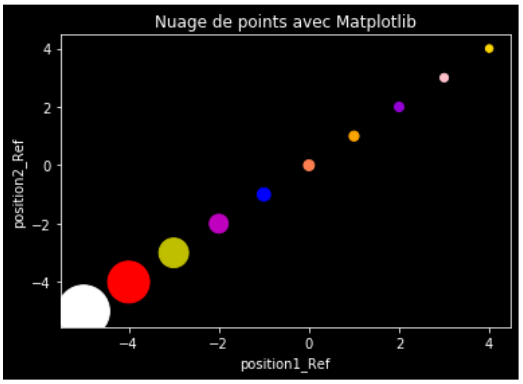
- **def categorie_devine(data, origine, k):**

Partie 2

Traitement des données data science

Utiliser le fichier **dataetoile_REf.cvs** comme référence dont la structure est la suivante : [position1_Ref,position2_Ref,etoile_Ref,temperature_Ref,masse_Ref]

1) Définir la liste DIM Correspondant à 30 fois masse_Ref et utiliser comme paramètre afin d'obtenir le graphique suivant



2) Reconnaitre le type de toutes les étoiles dans le fichier dataetoile à l'aide **k_proches_voisins et categorie_devine**
3) Donner la répartition des différents types d'étoiles sous forme de tableau (%)
4) En utilisant une fonction de tri par fusion, classer par ordre croissant les températures sans perdre les valeurs associées à chaque étoile.
5) Ecrire une fonction **data_etoile_tri** qui écrit 4) dans un fichier nouveau fichier data_etoile_tri .cvs
6) Pour chaque type d'étoile, définir la moyenne des masses et les températures associées.
   Définir la fonction **Pearson** qui retourne la valeur suivante :
   Coefficient de corrélation linéaire expliquer sont sens

$$R^2 = \frac{Covariance(L1, L2)}{EcartType(L1)(*\ EcartType(L2))}$$

7) Pour chaque type d'étoile, en utilisant la méthode des moindres carrés, donner l'équation de la régression linéaire et afficher R^2.
8) Faire une régression linéaire de deux listes contenant les moyennes de température et de masse de chaque type d'étoile associée.
9) Donner l'équation de la régression linéaire et afficher R^2.
10) Faire les graphiques suivants :

A. Température en fonction des masses. Puis
B. Log(Température/température) en fonction des log(masses).
C. Faire les graphiques en fonction des positions du type d'étoiles :
D. Et une analyse bilan