

La robotique et sa répercussion sur la société moderne

La répartition des emplois en fonction du phénomène de robotisation

Groupe

KRISNI Almehti
GOJAK Zlatan
MOISSINAC Xavier
BADRIOUCHE Mohamed

Sommaire

1. Introduction
2. La robotisation des transports
3. Le modèle de prévision statistique
4. Le modèle en grille et la variation active
5. Les entreprises fabricantes et l'économie industrielle
6. La modélisation finale
7. Conclusion

Introduction



Contexte de la recherche et de la modélisation

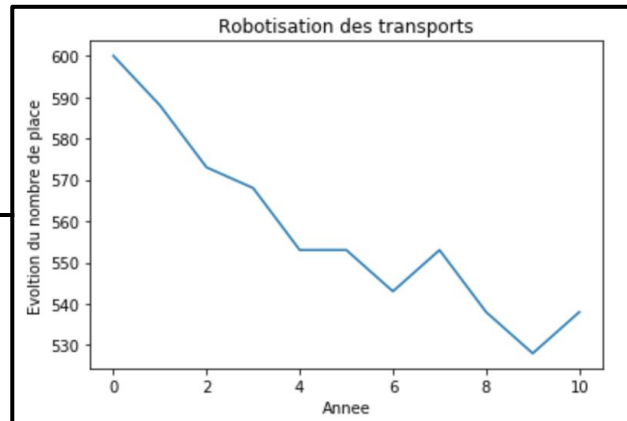
La robotisation des transports



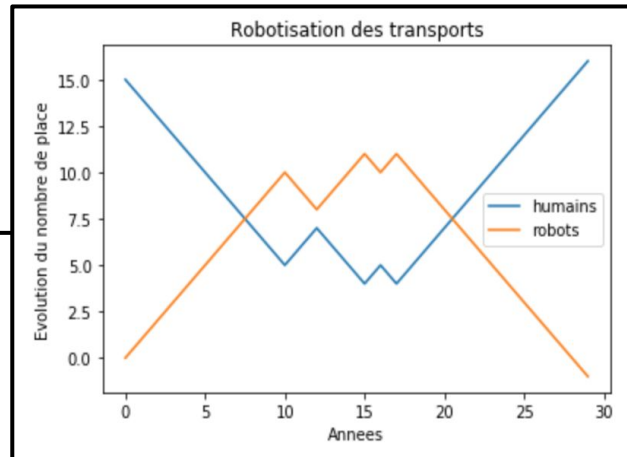
1. La prise en compte de paramètres simples
2. Un premier modèle

La prise en compte de paramètres simples

```
def remplacement(n ,prixr ,prihx ,nbreplaced ,esp ,rep):  
    #robot:list[int]  
    robot=[0]  
    #compteurr:int  
    compteur_r=0  
    #compteurh:int  
    compteur_h=nbreplaced  
    #humain:list[int]  
    humain=[nbreplaced]  
    #evoprix:list[int]  
    evoprix=evolution_prix(prixr,n)  
    #i:int  
  
    for i in range(n):  
        prixr = evoprix[i]  
        #ad:int  
        ad = avantage(prixr , prihx , esp , rep)  
        if (1>ad):  
            compteur_h = compteur_h-1 # diminue compteur humain  
            humain.append(compteur_h)  
            compteur_r = compteur_r + 1 # augmente compteur robot  
            robot.append(compteur_r)  
        else:  
            humain.append(compteur_h)  
            robot.append(compteur_r)  
    return (robot , humain)
```

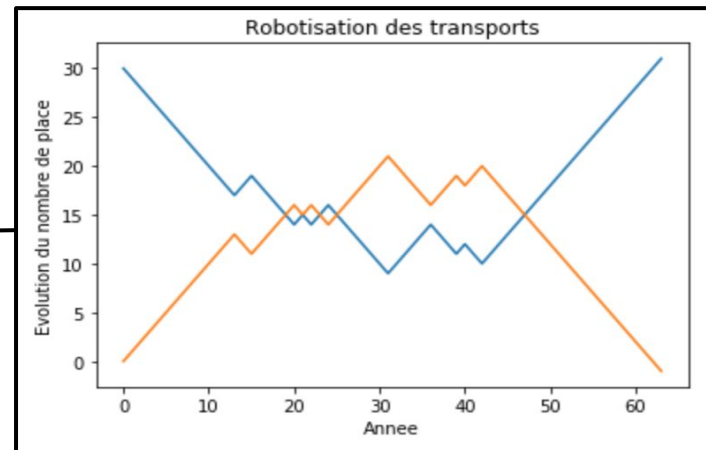


```
def new_remplacement(n ,prixr ,prihx ,nbreplaced ,esp ,rep):  
    #robot:list[int]  
    robot = [0]  
    #compteurr:int  
    compteur_r = 0  
    #compteurh:int  
    compteur_h = nbreplaced  
    #humain:list[int]  
    humain = [nbreplaced]  
    #evoprix:list[int]  
    evoprix = evolution_prix(prixr,n)  
    #i:int  
    i = 0  
  
    while ((i <= n) and (compteur_h >= 0) and (compteur_r >= 0)):  
        prixr=evoprix[i]  
        #ad:int  
        ad=avantage(prixr , prihx , esp , rep)  
        if (1>ad):  
            compteur_h = compteur_h-1 # diminue compteur humain  
            humain.append(compteur_h)  
            compteur_r = compteur_r + 1 # augmente compteur robot  
            robot.append(compteur_r)  
            i=i+1  
        else:  
            compteur_h = compteur_h + 1  
            humain.append(compteur_h)  
            compteur_r = compteur_r - 1  
            robot.append(compteur_r)  
            i=i+1  
    return (robot , humain)
```



Un premier modèle

```
def effet_crise(n ,prixr ,prihx ,nbreplaced ,esp ,rep , n2, n3):  
    #robot:list[int]  
    robot = [0]  
  
    #compteur:int  
    compteur_r = 0  
  
    #compteurh:int  
    compteur_h = nbreplaced  
  
    #humain:list[int]  
    humain = [nbreplaced]  
  
    #evoprix:list[int]  
    evoprix = evolution_prix(prixr,n)  
  
    #i:int  
    i = 0  
  
    while (i <= n2):  
        prixr=evoprix[i]  
  
        #ad:int  
        ad=avantage(prixr, prihx, esp, rep)
```



On peut modéliser une première répartition des emplois entre hommes et robots.

Néanmoins, elle reste simple et ne représente pas un modèle répondant à nos attentes.

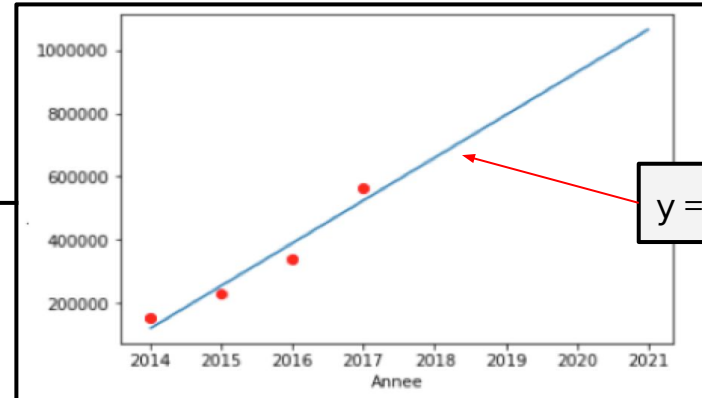
Le modèle de prévision statistique



1. La régression linéaire
2. La régression exponentielle

La régression linéaire

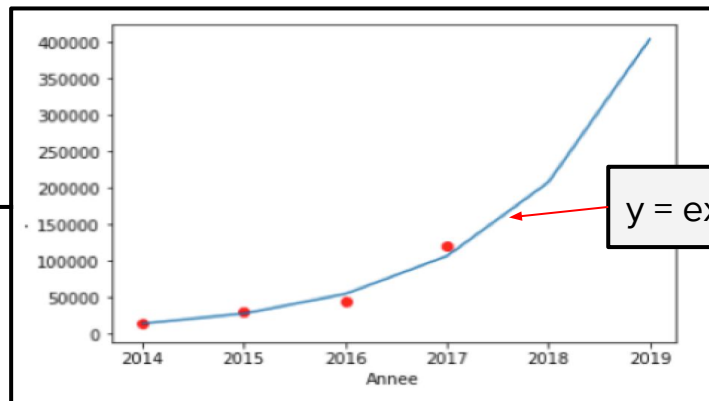
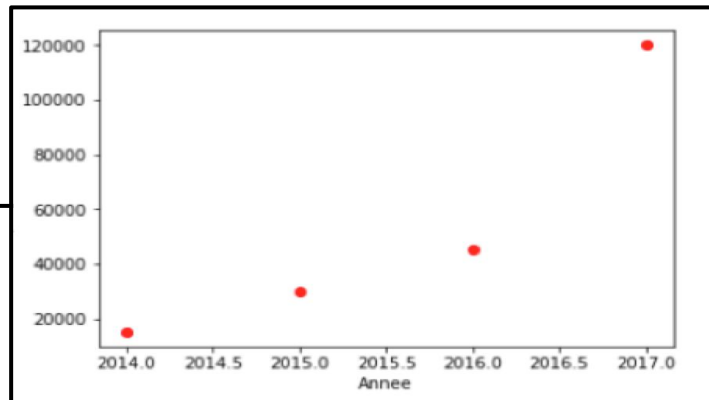
```
def evolution_lineaire_dico (D, annee_fin):  
    """dict[int:int] * int -> dict[int:float]"""  
    #a1 : number, b1 : number, i : int, a : float, b : float  
    #k : int, v : int  
    liste2=[]  
    newlistex=[]  
    i = 0  
    for k in D :  
        i = i+1  
        newlistex.append(i)  
        liste2.append(D[k])  
  
    a1 = np.array(newlistex)  
    b1 = np.array(liste2)  
    (a,b,rho,x,y)= linregress(a1,b1)  
  
    #k int, v : number, donnee : dict[int:float]  
    k = 0  
    j = annee_fin  
    donnee = dict()  
    while j > 0 :  
        k = k + 1  
        donnee[k] = round((a*k)+b, 2)  
        if (donnee[k] < 0 and (a*k) < 0) :  
            print("La dernière année est",k+2018)  
            return donnee  
  
        j = j-1  
    return donnee
```



$$y = (a * x + b)$$

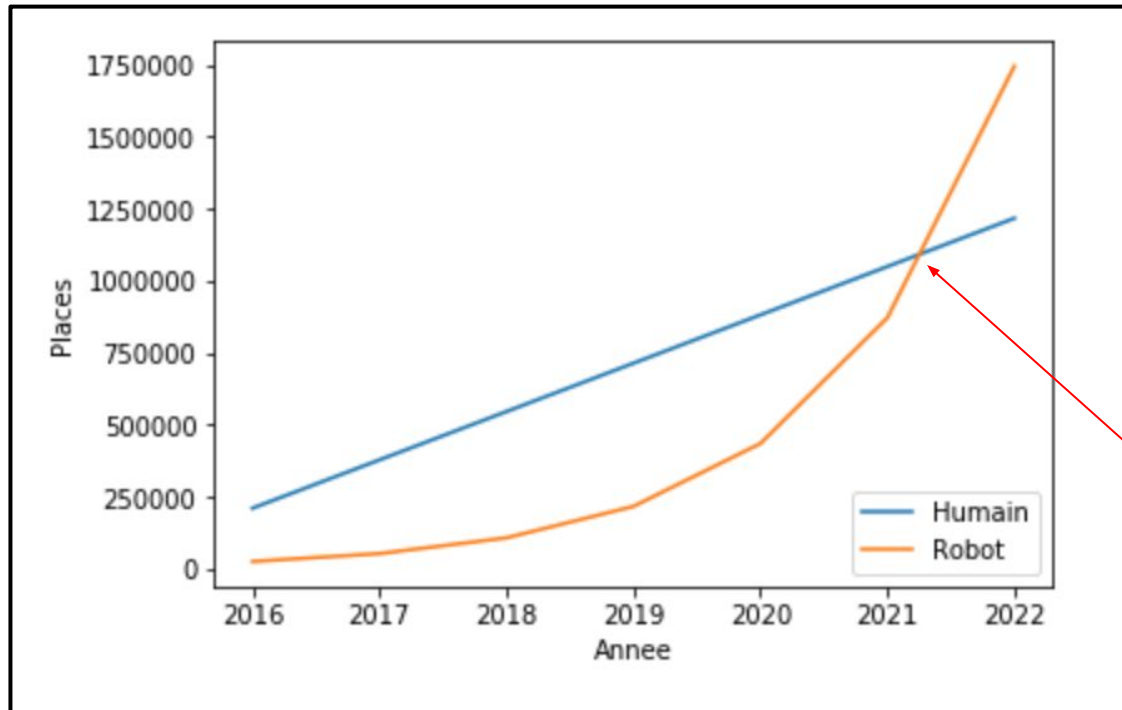
La régression exponentielle

```
def evolution_expo_dico_2 (D, annee_fin):  
    """dict[int:int] * int -> dict[int:float]  
    Hyp : Adapter à partir de la fonction polytfit d  
    #a1 : number, b1 : number, i : int, a : float, b : float  
    #k : int, v : int  
    liste2=[]  
    newlistex=[]  
    temp = 0  
    i = 0  
    for k in D :  
        i = i+1  
        newlistex.append(i)  
        liste2.append(D[k])  
        temp = k+1  
        temp2 = i+1  
  
    x = np.array(newlistex)  
    y = np.log(liste2)  
    (a,b) = np.polyfit(x, y,1)  
  
    #k int, v : number, donnee : dict[int:float]  
    k = 1  
    j = annee_fin  
    donnee = dict()  
    while j > 0 :  
        donnee[k] = round(np.exp(a*k)*np.exp(b),2)  
        k=k+1  
        temp2 = temp2 +1  
        j = j-1  
    return donnee
```



$$y = \exp(a * x + b)$$

Un cas d'étude



Le nombre de salariés et le nombre d'automates présents au sein de la compagnie Amazon

L'intersection arrive au début de l'an 2021.

Le modèle en grille et la variation active



1. Une vision plus pratique
2. La variation active des paramètres et sa conséquence

Une vision plus pratique

```
def rplcmt_homme_usine_grille (grid, lignes_grid, colonnes_grid, proba_rplcmt) :  
    # i : int  
    i = 0  
    # j : int  
    j = 0  
    # r : int  
    r = 0  
  
    while (i < lignes_grid) :  
        while (j < colonnes_grid) :  
            r = 100 - random.randint(0,100)  
            if (r < proba_rplcmt) :  
                # i2 : int  
                i2 = 0  
                # j2 : int  
                j2 = 0  
                for i2 in range(i-1,i+2) :  
                    for j2 in range(j-1,j+2) :  
                        if ((i2 >= 0) and (i2 < lignes_grid) and (j2 >= 0) and (j2 < colonnes_grid)) :  
                            if ((j2 == j) and (i2 == i)) :  
                                grid[i2][j2] = 'R'  
                            if (grid[i2][j2] != 'R') :  
                                grid[i2][j2] = '  
            i = i + 1  
            j = j + 1  
            i = i + 1  
            j = 0  
  
    return grid.copy()
```

Méthode d'affichage possédant avantages et inconvénients.

```
[ ['H' 'H' ' ' 'R' ' ' ' ' 'H' 'H' 'H' 'H' 'H' 'H' 'H' 'H' 'H']  
  ['H' 'H' ' ' ' ' 'R' ' ' 'H' 'H' 'H' 'H' 'H' 'H' 'H' 'H']  
  ['H' 'H' 'H' 'H' ' ' 'R' ' ' 'H' ' ' 'H' 'H' 'H' 'H' 'H']  
  ['H' 'H' 'H' 'H' 'H' 'H' 'H' ' ' 'H' ' ' 'H' 'H' 'H' 'H']  
  ['H' 'H' 'H' 'H' 'H' 'H' 'H' ' ' 'R' ' ' 'H' 'H' 'H' 'H']  
  ['H' 'H' 'H' 'H' 'H' 'H' 'H' ' ' 'H' ' ' 'H' 'H' 'H' 'H']  
  ['H' 'H' 'H' 'H' 'H' 'H' 'H' ' ' 'R' ' ' 'H' 'H' 'H' 'H']  
  ['H' 'H' 'H' 'H' 'H' 'H' 'H' ' ' 'H' ' ' 'H' 'H' 'H' 'H']  
  [' ' ' 'H' 'H' 'H' 'H' 'H' ' ' 'H' 'H' 'H' 'H' 'H' 'H']  
  ['R' ' ' 'H' ' ' ' ' 'H' 'H' 'H' 'H' 'H' 'H' 'H' 'H']  
  [' ' ' 'H' ' ' 'R' ' ' 'H' 'H' 'H' 'H' 'H' 'H' 'H']  
  [' ' ' ' ' ' ' 'H' 'H' 'H' 'H' 'H' 'H' 'H' 'H' 'H']  
  [' ' 'R' ' ' 'H' ' ' ' 'H' 'H' 'H' 'H' 'H' 'H']  
  [' ' ' 'H' ' ' 'R' ' ' ' ' 'H' 'H' 'H' 'H' 'H']  
  ['H' 'H' 'H' 'H' 'H' 'H' 'H' ' ' 'R' ' ' 'H' 'H']  
  ['H' 'H' 'H' 'H' 'H' 'H' 'H' 'H' 'H' 'H' ' ' 'R']  
  ['H' 'H' 'H' 'H' 'H' 'H' 'H' 'H' 'H' 'H' 'H' 'H']  
  ['H' 'H' 'H' 'H' 'H' 'H' 'H' 'H' 'H' 'H' 'H' 'H']  
  ['H' 'H' 'H' 'H' 'H' 'H' 'H' 'H' 'H' 'H' 'H' 'H']  
  ['H' 'H' 'H' 'H' 'H' 'H' 'H' 'H' 'H' 'H' 'H' 'H']  
  ['H' 'H' 'H' 'H' 'H' 'H' 'H' 'H' 'H' 'H' 'H' 'H']  
  ['H' ' ' 'R' ' ' 'H' 'H' 'H' 'H' 'H' 'H' 'H']  
  ['H' ' ' 'R' ' ' ' ' 'H' 'H' 'H' 'H' 'H' 'H']  
  ['H' 'H' ' ' ' ' ' ' 'R' ' ' 'H' 'H' 'H']
```

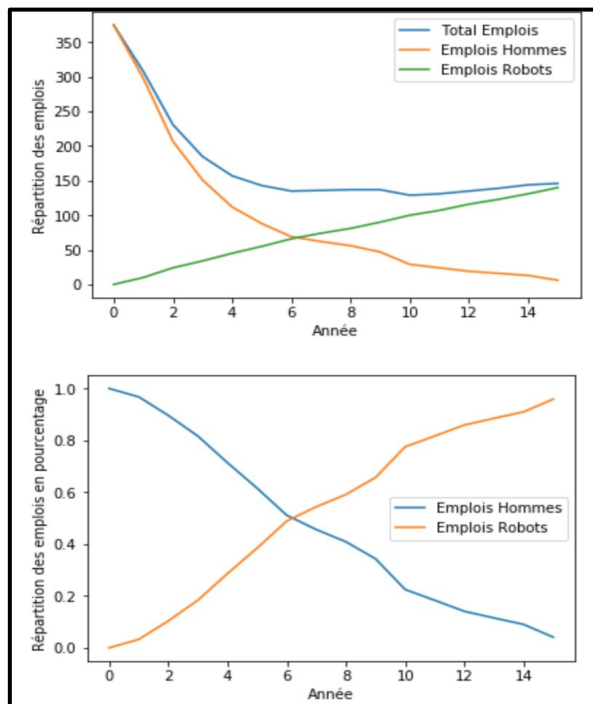
La variation active de paramètres

“Rien ne reste figé, tout est mouvement.”
- KRISNI Almehdi, 2019

```
def rplcmt_homme_usine_grille_n_années_dvlpmnt_robots (grid, lignes_grid, colonnes_grid, proba_rplcmt, n_années,  
  
    #grid_act = grid  
  
    # list_grid : dict[int:np]  
    dict_grid = dict()  
  
    dict_grid[0] = grid  
  
    proba_replacement = proba_rplcmt  
  
    for x in range(1,n_années+1) :  
  
        proba_replacement = proba_replacement + random.randint(0,dev_robot)  
        dict_grid[x] = rplcmt_homme_usine_grille(dict_grid[x-1].copy(),lignes_grid,colonnes_grid,proba_rplcmt)  
  
    return dict_grid
```

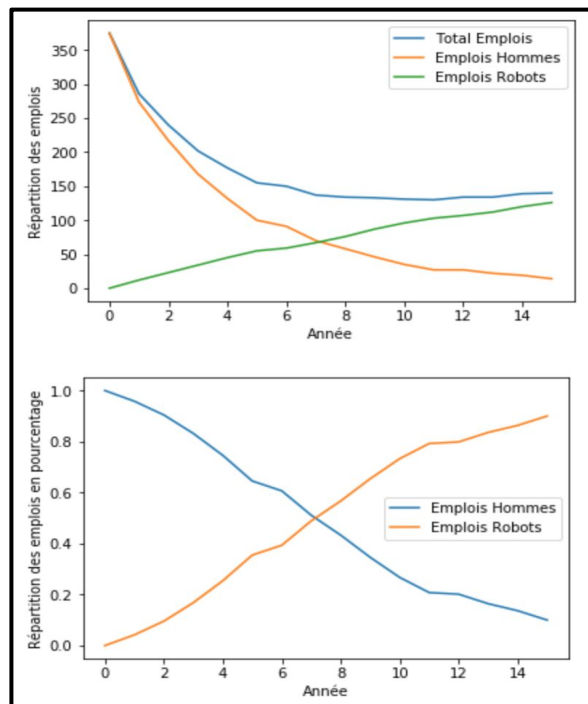
Afin de créer cette variation active, on permet à la variable “proba_rplcmt” d’augmenter de manière aléatoire

Des différences créées par la variation active



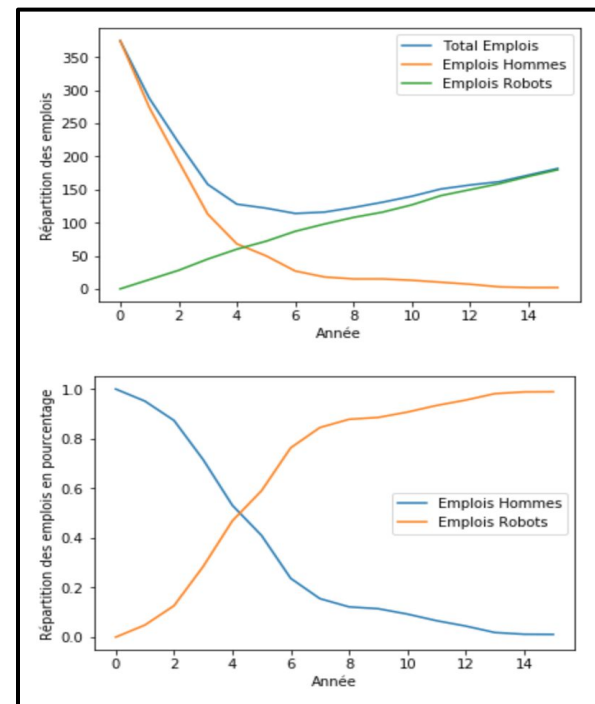
Aucune variation active :

- Probabilité fixée à 0.05
- Variation fixée à 0.00



Variation active faible :

- Probabilité fixée à 0.05
- Variation fixée à 0.02



Variation active forte :

- Probabilité fixée à 0.05
- Variation fixée à 0.05

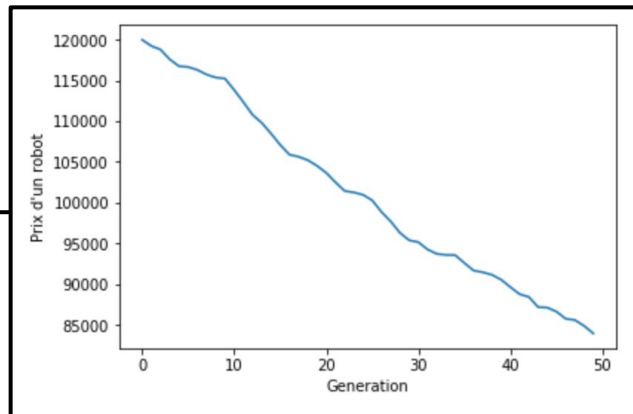
Les entreprises et l'économie sectorielle



1. L'évolution du capital d'une entreprise et du prix d'un robot
2. La clé d'un modèle plus complet et réaliste

L'évolution du prix d'un robot

```
def evolution_prix (robots_prix,nb_annees, min, max):  
    liste_robots_prix = list()  
    liste_robots_prix.append(robots_prix)  
  
    i = 0  
    while (i < nb_annees):  
        if (i < nb_annees//2):  
            min = min + 0.001  
            max = max + 0.001  
            coef_random = random.uniform(min,max)  
            liste_robots_prix.append(liste_robots_prix[i] - (liste_robots_prix[i])*coef_random//100)  
            i = i + 1  
        elif ( i >= nb_annees//2):  
            min = min  
            max = max - 0.004  
            coef_random = random.uniform(min,max)  
            liste_robots_prix.append(liste_robots_prix[i] - (liste_robots_prix[i])*coef_random//100)  
            i = i + 1  
  
    return liste_robots_prix
```



Chaque année, le prix du robot diminue d'environ 0 à 1.5% par rapport à l'année précédente.

L'évolution initiale du capital d'une entreprise

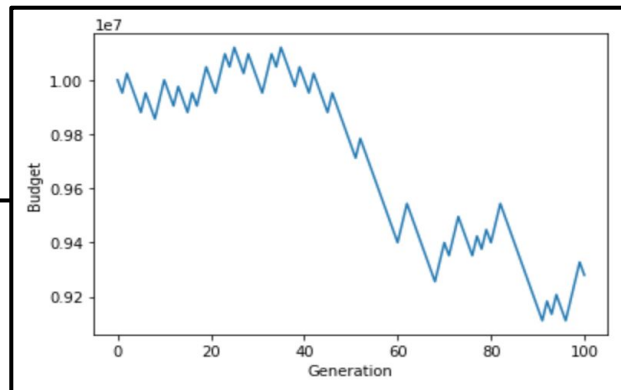
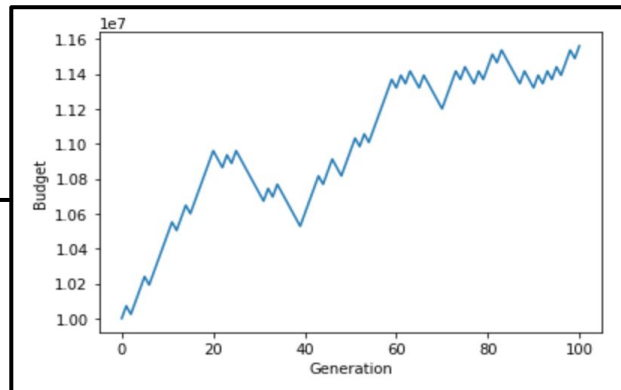
```
import random
def economie_entreprise (budget_initial,prix_robots, nb_annees, demande, concurrents):

    min = 0
    max = 1
    var_demande = demande
    var_concurrents = concurrents
    i = 0
    liste_budget = list()
    liste_budget.append(budget_initial)
    budget = budget_initial
    frais = prix_robots/100*40

    while (i != nb_annees):
        proba_vente = random.uniform((min+var_demande-var_concurrents),(max+var_demande-var_concurrents))
        if (proba_vente > 0.5):
            budget = budget + prix_robots - frais
            liste_budget.append(budget)
        else:
            budget = budget - frais
            liste_budget.append(budget)

        i = i + 1

    return liste_budget
```



La fonction finale

```
import random
def economie_entreprise_Final_Marketing (budget_initial,prix_robots, nb_annees, demande, concurrents):

    min = 0
    max = 1
    var_demande = demande
    var_concurrents = concurrents
    var_prix_robots = prix_robots
    i = 0
    liste_budget = list()
    liste_budget.append(budget_initial)
    budget = budget_initial
    frais_marketing = var_prix_robots//10
    frais = prix_robots/100*30 + frais_marketing
    liste_demande = list()
    liste_concurrents = list()

    while (i != nb_annees):
        proba_vente = random.uniform((min+var_demande-var_concurrents),(max+var_demande-var_concurrents))
        if (proba_vente > 0.5):
            budget = budget + var_prix_robots - frais
            liste_budget.append(budget)
        else:
            budget = budget - frais
            liste_budget.append(budget)

        i = i + 1

        random_var = random.uniform(0,100)
        if random_var > 80:
            var_demande = var_demande + 0.02
        elif random_var < 20:
            var_concurrents = var_concurrents + 0.02
        elif random_var < 1:
            frais = frais - 0.3 * frais

        min1 = 0
        max1 = 1.5

        if (i<nb_annees//2):
            min1 = min1 + 0.001
            max1 = max1 + 0.001
            coef_random = random.uniform(min1,max1)
            var_prix_robots = var_prix_robots - var_prix_robots*coef_random//100
        elif (i >= nb_annees//2):
            min1 = min1
            max1 = max1 + 0.1
            coef_random = random.uniform(min1,max1)
            var_prix_robots = var_prix_robots - var_prix_robots*coef_random//100

        if (proba_vente < 0.5):
            frais_marketing = frais_marketing + var_prix_robots//100
            var_demande = var_demande + 0.01

    return liste_budget
```

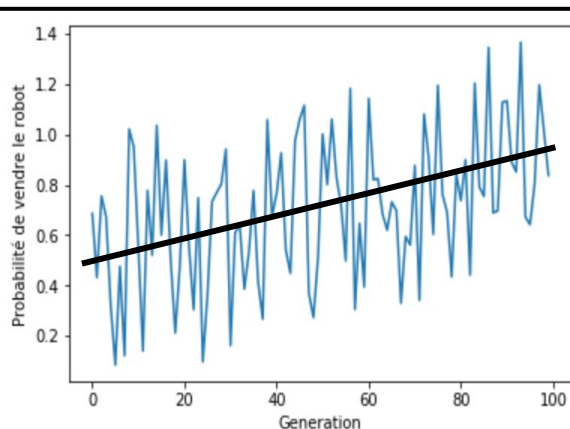
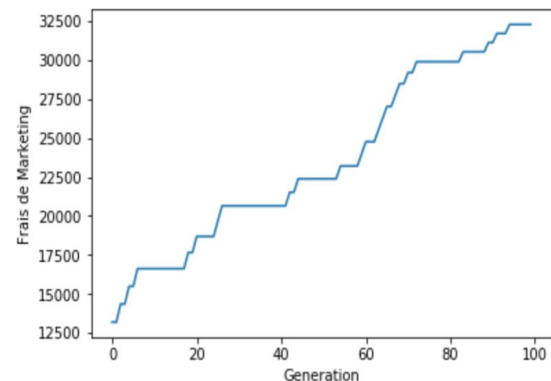
Initialisations

Variation du
capital de base

Variation de
la demande

Evolution du
prix d'un robot

Evolution des
frais liés au
marketing



La modélisation finale



Une combinaison gagnante

Une combinaison gagnante

Suite aux modélisations précédentes, on ne retiendra que certains éléments utilisés précédemment.

On retiendra la probabilité active, les différentes formes de régression, le capital d'une entreprise et le prix du robot.

En ce qui concerne l'affiche, l'utilisation de la grille est considérée comme non-pratique et on retiendra une évolution statistique simple modélisée par des graphiques.



On a donc une idée en tête ...

Conclusion



Qu'avons-nous appris suite à tout ce travail ?
Peut-on réellement prédire une telle situation ?