Projet de fin de 2éme année
Filière **Génie logiciel**

Implémentation d'un modèle prédictif pour la classification des tumeurs

Soutenu par :

Hamzaoui Aiman

Ben Ayad Mohamed Ayoub

Encadré par :

El Asri Bouchra

Jury:

Azzedine El hassouny

Année universitaire 2017-2018

PLAN

- Introduction
- Contexte général du projet
- Démarche
- Définition et analyse de données
- Mise en œuvre de la solution
- Conclusion

Introduction

Contexte général du projet

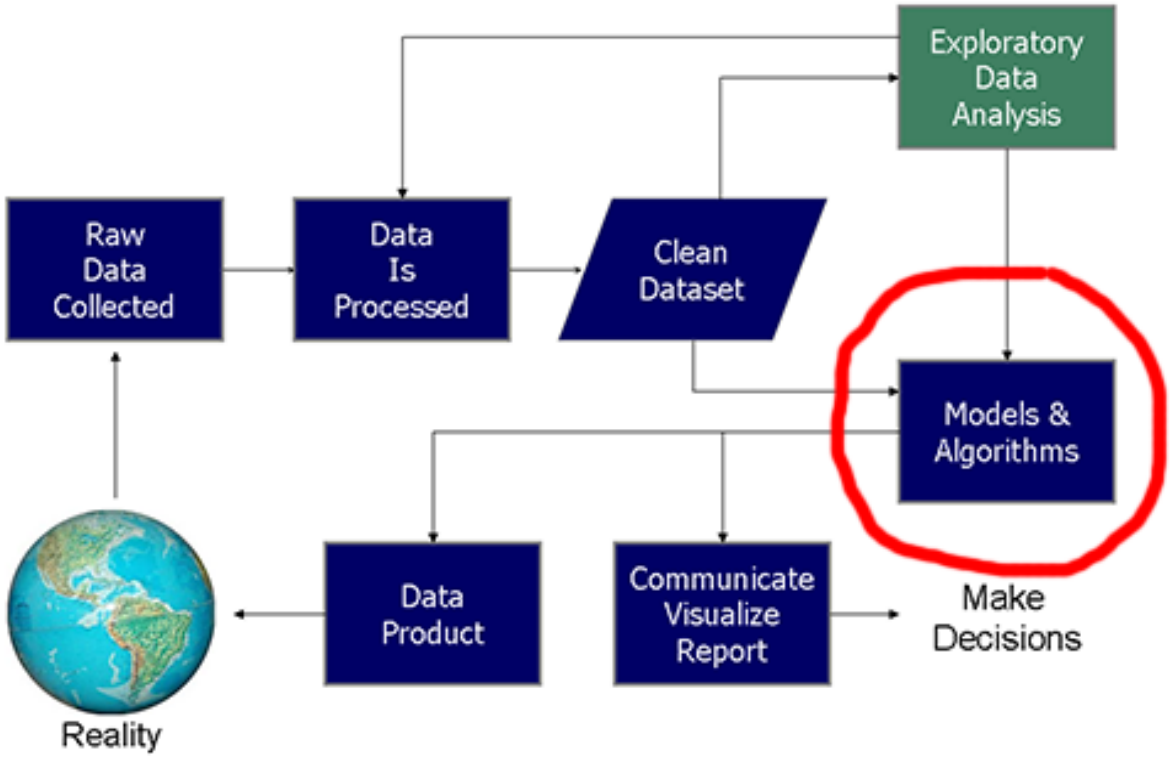
Big Data et sante :

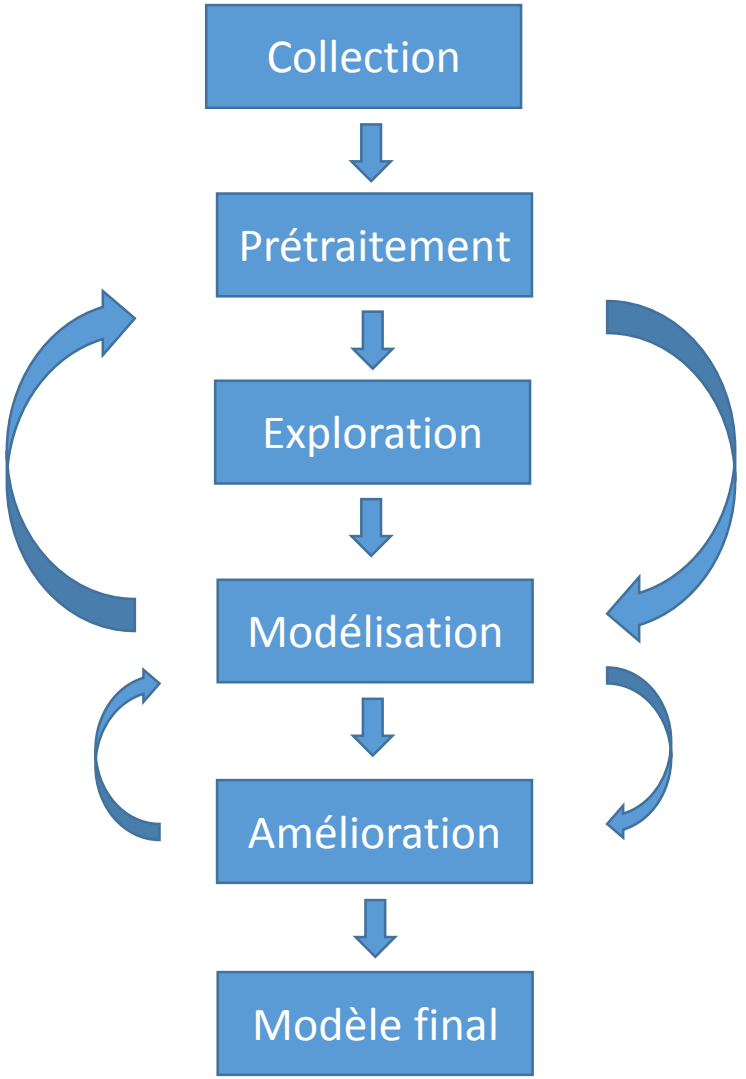
- Les outils Big Data ont permis le lancement de nombreux projets médicaux fondés sur l'exploitation de données massives.

Le but de notre travail est de concevoir un système d'aide aux médecins radiologues. Pour ce faire, nous devrons détecter des tumeurs cancéreuses à partir d'image numérisée d'une aspiration d'aiguille fine (FNA) d'une masse mammaire.

Démarche

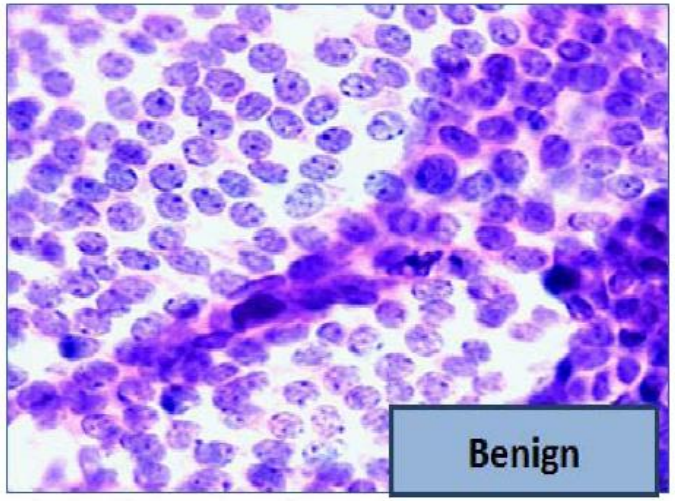
Les étapes

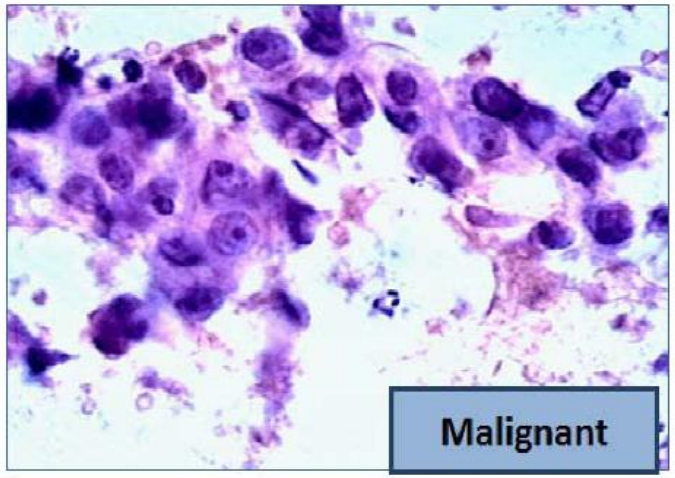

Exploratory Data Analysis
Raw Data Collected
Data Is Processed
Clean Dataset
Models & Algorithms
Data Product
Communicate Visualize Report
Make Decisions
Reality


Collection
Prétraitement
Exploration
Modélisation
Amélioration
Modèle final

Définition et analyse de données

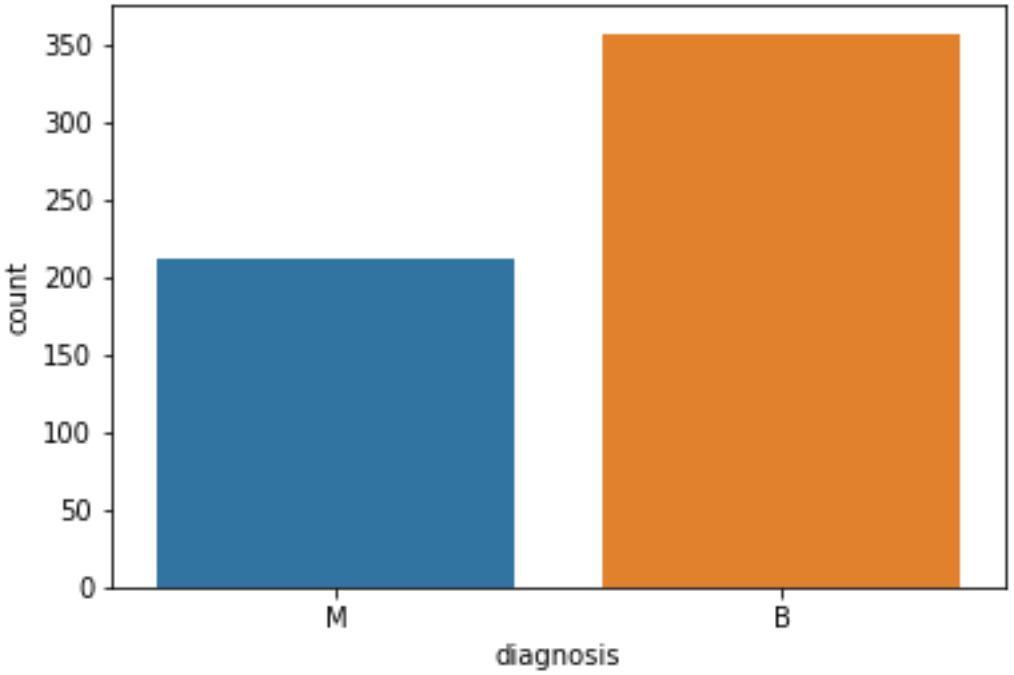
- La base de données ?
- Les dimensions ?
- Les caractéristiques ?
- Signification des données ?





	id	diagnosis	radius_mean	texture_mean	perimeter_mean	area_mean	smoothness_mean	compactness_mean	concavity_mean
0	842302	M	17.99	10.38	122.80	1001.0	0.11840	0.27760	0.3001
1	842517	M	20.57	17.77	132.90	1326.0	0.08474	0.07864	0.0869
2	84300903	M	19.69	21.25	130.00	1203.0	0.10960	0.15990	0.1974
3	84348301	M	11.42	20.38	77.58	386.1	0.14250	0.28390	0.2414
4	84358402	M	20.29	14.34	135.10	1297.0	0.10030	0.13280	0.1980

- Nous devons retirer les colonnes id , diagnosis avant l'apprentissage
- Nous devons supprimer la colonne id32 car ses valeurs sont nuls



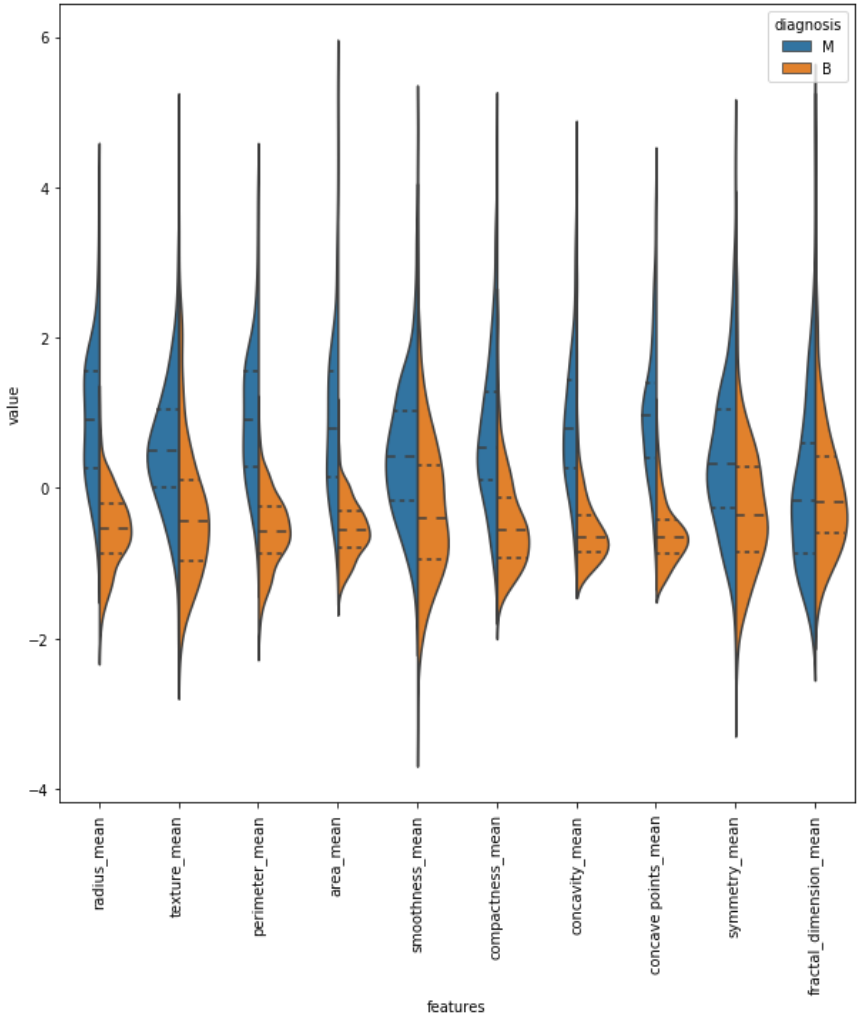
	id	radius_mean	texture_mean	perimeter_mean	area_mean	smoothness_mean	compactness_mean	concavity_mean	concave points_mean
count	5.690000e+02	569.000000	569.000000	569.000000	569.000000	569.000000	569.000000	569.000000	569.000000
mean	3.037183e+07	14.127292	19.289649	91.969033	654.889104	0.096360	0.104341	0.088799	0.048919
std	1.250206e+08	3.524049	4.301036	24.298981	351.914129	0.014064	0.052813	0.079720	0.038803
min	8.670000e+03	6.981000	9.710000	43.790000	143.500000	0.052630	0.019380	0.000000	0.000000
25%	8.692180e+05	11.700000	16.170000	75.170000	420.300000	0.086370	0.064920	0.029560	0.020310
50%	9.060240e+05	13.370000	18.840000	86.240000	551.100000	0.095870	0.092630	0.061540	0.033500
75%	8.813129e+06	15.780000	21.800000	104.100000	782.700000	0.105300	0.130400	0.130700	0.074000
max	9.113205e+08	28.110000	39.280000	188.500000	2501.000000	0.163400	0.345400	0.426800	0.201200

8 rows × 32 columns

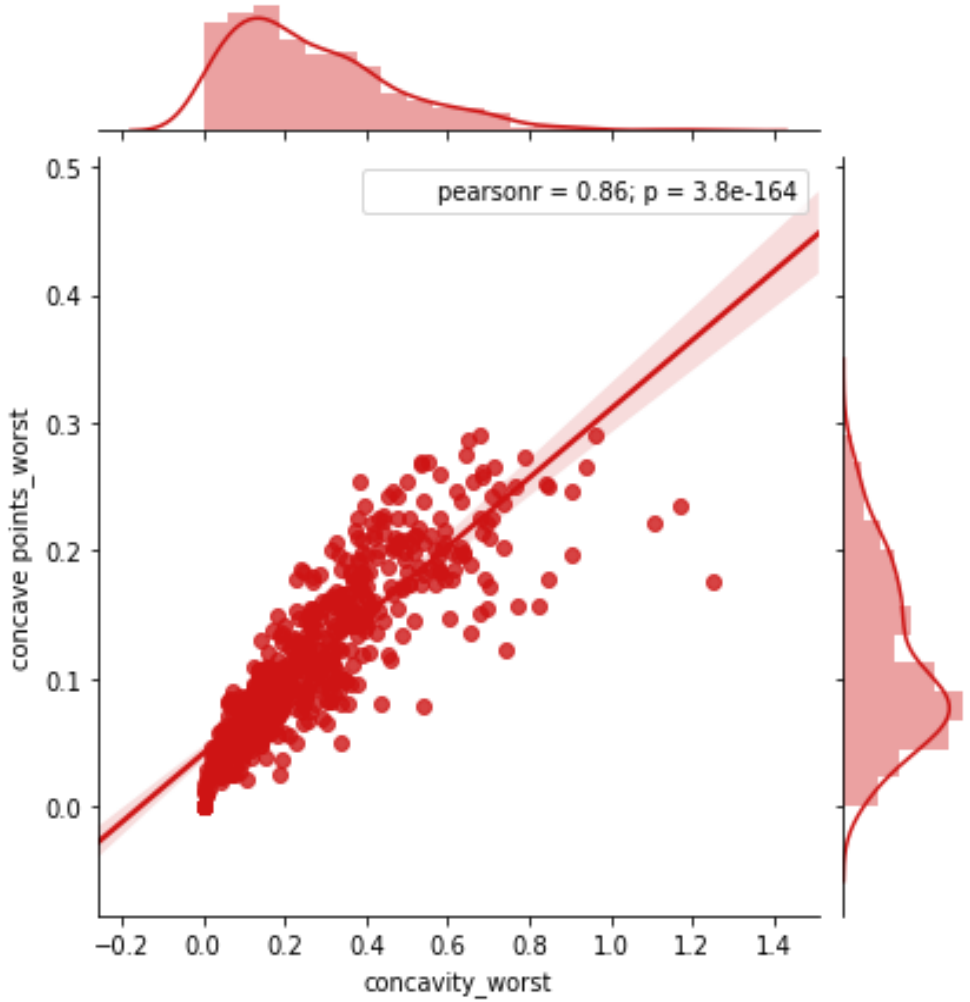
- Les tumeurs bégnines sont plus nombreuses que les tumeurs malignes
- Les données ne sont pas normalisées ni standardisées

- Pourquoi la visualisation ?
- Violin plot ?
- Hypothèses:

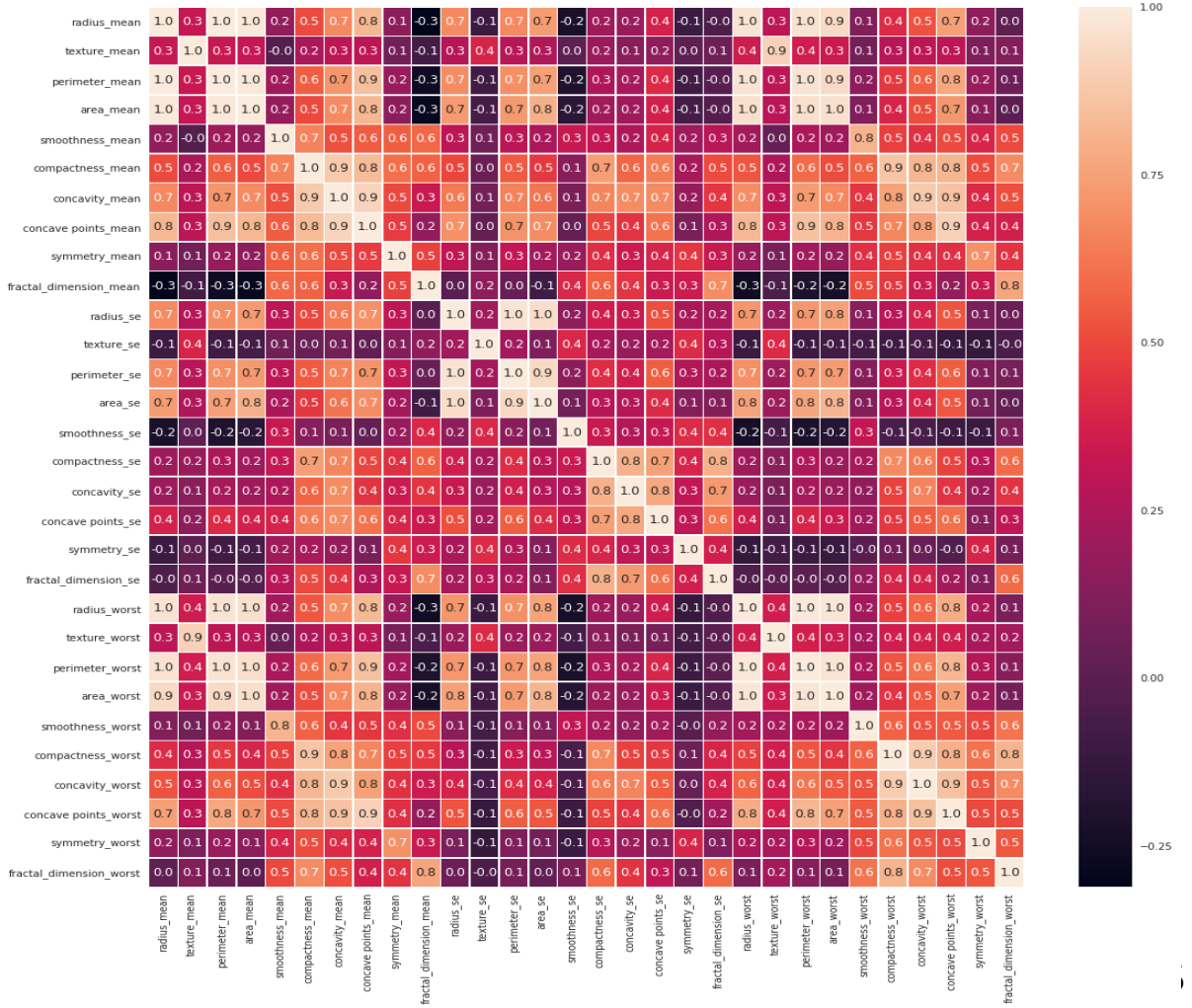
  - Les deux dernières variables ont à peu près la même distribution, elles peuvent être corrélées
  - Les médianes des catégories de la variable **radius_mean** sont écartées, cette variable peut être bonne pour la classification



- Joint plot ?
- Hypothèses :
  - Les deux variables semblent corrélées.



- Correlation map ?
- Affirmation :
  - ➢ Ce graphe nous permet de visualiser les corrélations entre les variables et de valider ou de rejeter les hypothèses émises par les différentes techniques de visualisation concernant les groupes qui semblaient être corrélés.



.

- Pourquoi la sélection :

La sélection de caractéristiques est une technique permettant de choisir les caractéristiques les plus pertinentes, celles adaptées à la résolution d'un problème particulier, pour minimiser la complexité de calcule et éviter la redondance des données.

- **Corrélation**
- **Univariate feature selection**
- **Recursive feature selection**
- **Tree based feature selection**

Mise en œuvre de la solution

- Le choix des algorithmes ?
- Pourquoi ses algorithmes ?
- Comment tester ses algorithmes ?

- Résultats ?
- Résultats biaises ?

Toutes les VARIABLES

```
print("Score de SVM : ",classification_accuracy(x,y,svm.SVC(),x.columns))
```

```
Score de SVM :  0.9736842105263158
```

```
print("Score de RandomForrest",classification_accuracy(x,y,RandomForestClassifier(n_estimators=100),x.columns))
```

```
Score de RandomForrest 0.9780701754385965
```

```
print("Score de la Regression Logistique :",classification_accuracy(x,y,LogisticRegression(),x.columns))
```

```
Score de la Regression Logistique : 0.9736842105263158
```

TOP 6 des VARIABLES

```
print("Score de SVM : ",classification_accuracy(x,y,svm.SVC(),carac_plus_signifiantes))
```

```
Score de SVM :  0.9605263157894737
```

```
print("Score de RandomForrest",classification_accuracy(x,y,RandomForestClassifier(n_estimators=100),carac_plus_signifiante
```

```
Score de RandomForrest 0.9692982456140351
```

```
print("Score de la Regression Logistique :",classification_accuracy(x,y,LogisticRegression(),carac_plus_signifiantes))
```

```
Score de la Regression Logistique : 0.9824561403508771
```

- Pourquoi cette étape ?
- Quelles méthodes utilisées pour l'évaluation et pour l'amélioration ?

- Résultats

TOUTES les VARIABLES

```
print("Score de validation croisee du SVM : ",classification_accuracy_CV(x,y,svm.SVC(),x.columns))
```

```
Score de validation croisee du SVM :  0.9718832479428661
```

```
print("Score de RandomForrest",classification_accuracy_CV(x,y,RandomForestClassifier(n_estimators=100),x.columns))
```

```
Score de RandomForrest 0.9526005278683435
```

```
print("Score de la Regression Logistique :",classification_accuracy_CV(x,y,LogisticRegression(),x.columns))
```

```
Score de la Regression Logistique : 0.9771774569166279
```

TOP 6 des VARIABLES

```
print("Score de validation croisee du SVM : ",classification_accuracy_CV(x,y,svm.SVC(),carac_plus_signifiantes))
```

```
Score de validation croisee du SVM :  0.9419965843813072
```

```
print("Score de RandomForrest",classification_accuracy_CV(x,y,RandomForestClassifier(n_estimators=100),carac_plus_signifia
```

```
Score de RandomForrest 0.9525539512498058
```

```
print("Score de la Regression Logistique :",classification_accuracy_CV(x,y,LogisticRegression(),carac_plus_signifiantes))
```

```
Score de la Regression Logistique : 0.9473063188945815
```

- Résultats

SVM

```
model=svm.SVC()
param_grid = [

              {'C': [1, 10, 100, 1000],
               'gamma': [0.001, 0.0001],
               'kernel': ['rbf','linear']
              }
 ]
Classification_model_gridsearchCV(model,param_grid,carac_plus_signifiantes)
```

```
The best parameter found on development set is : {'C': 100, 'gamma': 0.001, 'kernel': 'linear'}
the bset estimator is  SVC(C=100, cache_size=200, class_weight=None, coef0=0.0,
  decision_function_shape='ovr', degree=3, gamma=0.001, kernel='linear',
  max_iter=-1, probability=False, random_state=None, shrinking=True,
  tol=0.001, verbose=False)
The best score is  0.9560632688927944
```

RandomForest

```
model = RandomForestClassifier()
param_grid = {
 'n_estimators': [200, 400, 600, 800, 1000]}
Classification_model_gridsearchCV(model,param_grid,carac_plus_signifiantes)
```

```
The best parameter found on development set is : {'n_estimators': 400}
the bset estimator is  RandomForestClassifier(bootstrap=True, class_weight=None, criterion='gini',
            max_depth=None, max_features='auto', max_leaf_nodes=None,
            min_impurity_decrease=0.0, min_impurity_split=None,
            min_samples_leaf=1, min_samples_split=2,
            min_weight_fraction_leaf=0.0, n_estimators=400, n_jobs=1,
            oob_score=False, random_state=None, verbose=0,
            warm_start=False)
The best score is  0.9543057996485061
```

Regression Logistique

```
model = LogisticRegression()

penalty = ['l1', 'l2']


C = np.logspace(0, 4, 10)


param_grid = dict(C=C, penalty=penalty)

Classification_model_gridsearchCV(model,param_grid,carac_plus_signifiantes)
```

```
The best parameter found on development set is : {'C': 2.7825594022071245, 'penalty': 'l1'}
the bset estimator is  LogisticRegression(C=2.7825594022071245, class_weight=None, dual=False,
          fit_intercept=True, intercept_scaling=1, max_iter=100,
          multi_class='ovr', n_jobs=1, penalty='l1', random_state=None,
          solver='liblinear', tol=0.0001, verbose=0, warm_start=False)
The best score is  0.9595782073813708
```

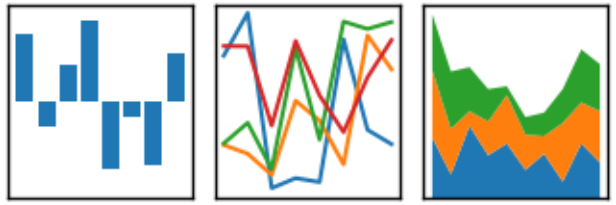
- Modèle final : **Régression Logistique**
  - **C**=2.78 // **penalty**='l1'
  - **Score = 96%**

jupyter

scikit learn

pandas

$$y_{it} = \beta' x_{it} + \mu_i + \epsilon_{it}$$



Conclusion

Merci pour votre attention