Introduction à l'Apprentissage Statistiques

Projet : Titanic
Encadré par : François LANDES
Membres de l'équipe : Gabriel UZAN, Ulukbek OMOEV, Thi Thuy Trang NGUYEN

I. Introduction

Dans le cadre de troisième année du cycle Informatique (UE Introduction à l'Apprentissage Statistique) à l'Université Paris Saclay, il nous est proposé un projet. Nous avons décidé de travailler sur le projet Titanic dont l'objectif est de répondre à la question : **Survivrez-vous à la catastrophe du Titanic ?**

En commençant par parler un peu sur le choix du dataset sur lequel on voudrait travailler, on voulait au tout début faire des prédictions concernant la reprise de la pandémie (COVID-19) vu la réalité qu'on a vécu des années dernières. Puis, suite au retour du pré-rapport, en raison de l'ambition/niveau de difficulté de notre idée, on devait changer le dataset au moment où on était censé de commencer notre projet. Ainsi, on s'est précipité de nous adresser notre professeur référent pour trancher sur le dataset sur quoi étudier pour le projet. Après plusieurs échanges de courriels, hésitations, propositions, on a fini par partir avec le set de données de Titanic, ce qui est un événement historique très populaire dans le monde entier. Par suite, avec ce data set et les connaissances qu'on pouvait accumuler en cette UE, on cherche à prédire si une personne peut survivre ou non de l'accident en question. Selon note professeur de référent, malgré le fait que le dataset n'est pas très avancé au niveau de préprocessing/nettoyage, que la taille du dataset est assez modeste. D'autre part on avait opté pour ce dataset en espérant qu'il serait pédagogique pour notre projet, c'est-à-dire qu'on pourrait prendre le projet de Titanic pour une opportunité de nous acquérir des connaissances de base en apprentissage statistique ainsi qu'une révision de ce qu'on avait été appris tout au long du semestre. Ainsi, dans la suite du rapport, on va commencer par décrire le dataset, le découvrir/étudier dans l'étape de Visualisation, le nettoyer en Préprocessing tout en rendant utilisable avant de le passer aux modèles prédictifs qu'on a choisi.

- ***Description du dataset***

On a téléchargé notre dataset depuis le site www.kaggle.com. Les données sont déjà séparées en deux fichiers : « train.csv » qui sert à la partie de l'entraînement et « test.csv » pour les tests après cet entraînement. Dans le fichier « train.csv », il y a 891 lignes, ce qui équivaut à 891 passagers, alors qu'il y en 418 dans le fichier « test.csv ». Puis, il y a 12 attributs dans « train.csv » et il y en a 11 dans « test.csv ». La colonne/l'attribut enlevé(e) dans « test.csv » est celle de « Survived », celle qui indique si une personne est survécu ou pas de l'événement Titanic. Ainsi, on cite ici une brève description/signification et les types des attributs de notre data set.

- **Passengerid** (type : int) : Numéro d'identification de passager

- **Survived** (type : int) : Le passager a survecu ou non? 0 = le passager est non survecu, 1 = le passager est survecu

- **Pclass** (type : int) : Classe de voyage. 1 = Première classe, 2 = Seconde classe, 3 = Troisième classe

- **Name** (type : objet) : Le nom du passager

- **Sex** (type : objet) : Le sexe du passager. male = homme, female = femme

- **Age** (type : float) : L'age du passager.

- **SibSp** (type : int) : Le nombre de frère/soeur/conjoint

- **Parch** (type : int) : Le nombre de parents-enfants

- **Ticket** (type : objet) : Le numéro du ticket

- **Fare** (type : float) : Le tarif du ticket

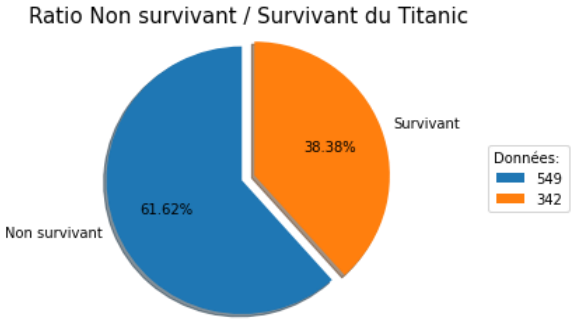
- **Cabin** (type : objet) : Le numéro du cabine

- **Embarked** (type : objet) : La ville où le passager a monté au bord du navire. C=Cherbourg, Q=Queenstown, S=Southampton

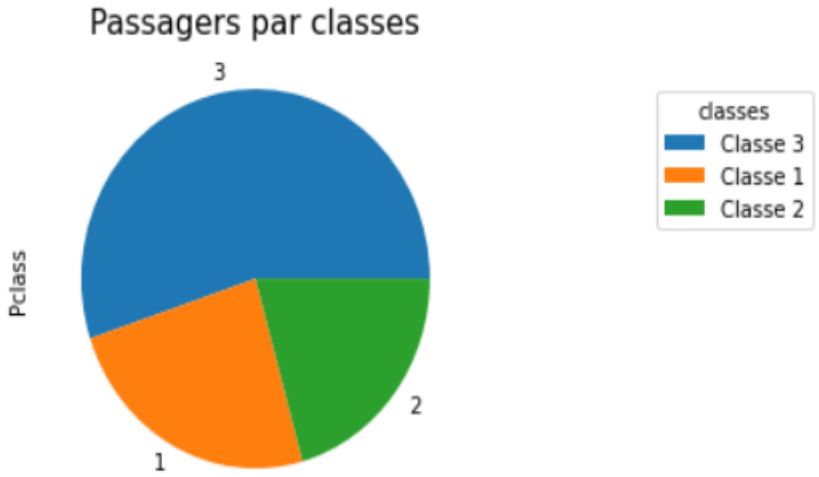
II. Visualisation

La visualisation est un outil très important dans le domaine de l'Apprentissage Automatique. Elle permet de tirer rapidement des informations grâce aux représentations graphiques. Sans faire du préprocessing ni analyse prédictive seulement en utilisant les méthodes et les fonctions des librairies matplotlib et seaborn nous avons pu remarquer des choses suivantes :

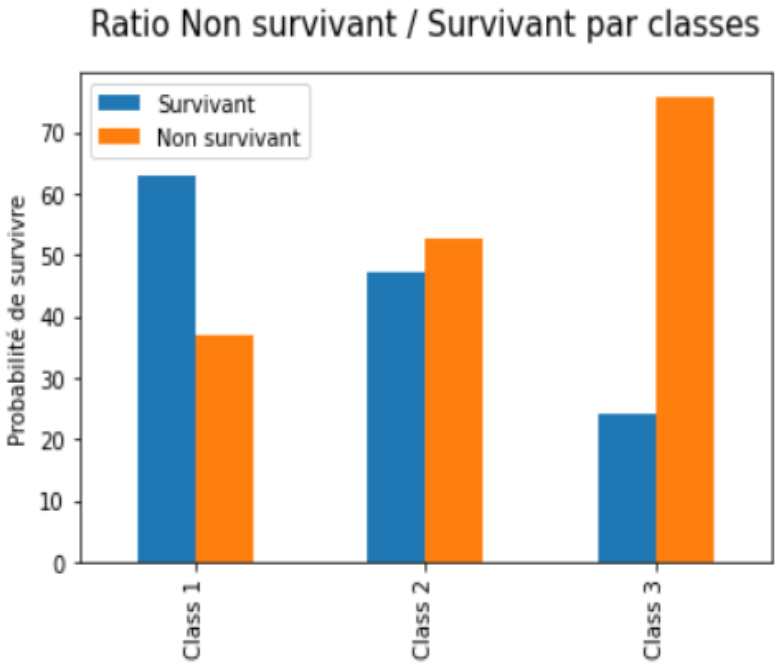
- Parmi 891 passagers 549 personnes ne sont pas survécu dans la catastrophe ce qui représente plus de 60% de la population.



- Les passagers du Titanic ont voyagé en 3 classes. La plupart des passagers ont voyagé dans la classe 3. Le nombre de passagers dans la classe 3 dépassait le nombre de passagers de classe 1 et de classe 2.
- Il y avait plus de chance de survivre si la classe était plus élevée.



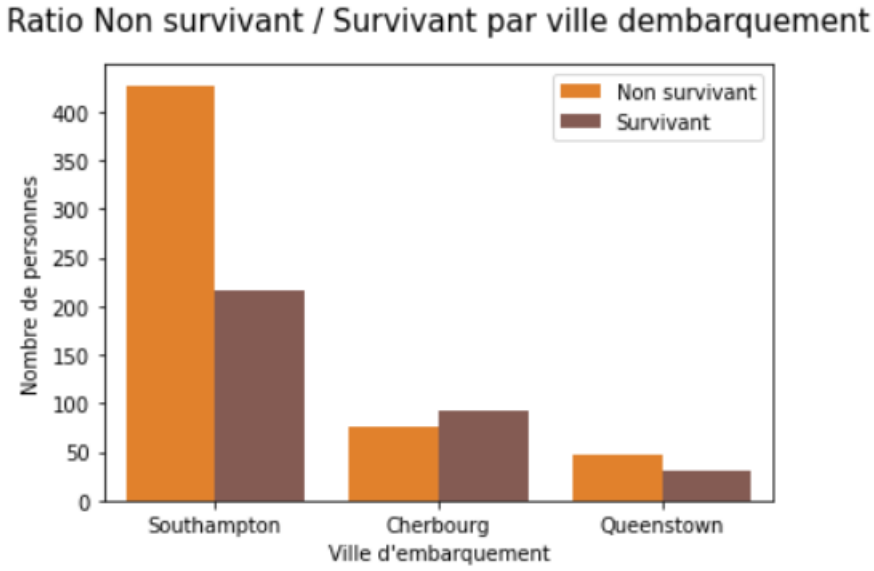
- Dans le diagramme suivant, on peut clairement voir que le taux de non survivant est proportionnel avec le numéro de classe. Ce qui est plus ou moins cohérent avec un scénario en vrai vie.



En plus, on a remarqué aussi qu'une femme a plus de chance de survivre contre un homme. Une approximation de 74% contre 19% peut prouver cela. On peut affirmer que les femmes/filles auraient été plus privilégiées au niveau du sauvetage en comparant avec les hommes/garçons.

	Sex	Survived
0	female	0.742038
1	male	0.188908

La plupart des voyageurs sont des jeunes car l'âge moyen du voyageur est légèrement inférieur à 30 ans. La plupart des enfants dont l'âge est inférieur à 10 ans ont survécus. Entre les passagers, la plupart des personnes ne se connaissaient pas. Car selon ce qui reflètent l'attribut « SibSp » et « Parch », seulement 8 personnes voyageaient avec leur frère/sœur/conjoint et au plus 6 parents/enfants voyageaient ensemble. Un autre élément qu'on a pu découvrir à travers de l'attribut « Embarked » c'est que les personnes qui se sont embarquées à Cherbourg avaient plus de chance de survivre que s'ils débarqueraient sur Southampton ou sur Queentown.



III. Préprocessing

Les algorithmes de Machine Learning apprennent à partir des sonnées qui leur sont fournis. Par conséquent si ces données sont de mauvaise qualité, qu'elles sont erronées, incomplètes ou redondantes alors l'algorithme qui en résulte sera lui-meme assez mauvais. Puisqu'il a censé refléter ce qu'il voit dans les données. Pour cette raison il est impératif de bien préparer nos données avant leur passage dans la machine. Il faut les nettoyer, les filtrer, les normaliser et ce qu'on va faire dans cette étape.

Pour notre dataset nous avons utilisé deux opérations de préprocessing :

- L'encodage (qui consiste à convertir les données qualitatives en valeur numérique).
- L'imputation (qui permet de remplacer des données manquantes par certaines valeurs statistiques).

L'opération d'encodage a été appliqué pour l'attribut « Sexe » qui indique le sexe du passager male/female ce qui a été remplacé par les valeurs numériques 0 et 1 respectivement. En ce qui concerne l'attribut « Embarked » qui indique la ville où le passager a monté au bord du navire, les villes Southampton, Queenstown et Cherbourg ont été remplacé par 0, 1 et 2.

L'opération d'imputation nous avons appliqué pour l'attribut « Age ». Les valeurs manquantes ont été remplacées par l'age moyen de tous les passagers.

Et enfin dans l'attribut « Embarked » il manquaient deux valeurs. On les a remplacé par la mode.

On a décidé d'éliminer les attributs qui sont à notre avis, ne contribuent pas à l'étude des données. Ce sont : «PassengerId », « Name », « Ticket », « Cabin », « Fare », « SibSp », « Parch ».

IV. Modèle prédictive

Et finalement, on a cerné le problème, on a récupéré les données et les a explorés, on en a extrait un jeu d'entraînement et un jeu de test, et on a programmé des « plus ou moins » pipelines de transformation pour nettoyer et préparer automatiquement des données pour les algorithmes d'apprentissage automatique. On va maintenant, enfin, être prêt à sélectionner et à entraîner nos modèles de Machine Learning choisis.

Pour cela, on a opté pour trois modèles « RandomForestClassifier() », « LogisticRegression(), LinearRegression() ». Et puis, on a également essayé avec la « Cross Validation » en espérant de voir le changement de taux d'exactitude (« Mean of accuracy ») des modèles en question.

```python
def cross_val_model(model, x, y):#____trg  doesnt work?
    cv = RepeatedStratifiedKFold(n_splits=10, n_repeats=3, random_state=1)
    scores = cross_val_score(model, x, y, scoring='accuracy', cv=cv, n_jobs=-1)
    return scores

def accuracy_model(model, x_train, y_train, x_test):
    model.fit(x_train,y_train)
    y_test=model.predict(x_test)
    print("Accuracy:",metrics.accuracy_score(y_test, y_pred))
    return metrics.accuracy_score(y_test, y_pred)

def accuracy_model_bis(model, x_train, y_train, x_test):
    print(model)
    model.fit(x_train, y_train)
    y_test = model.predict(x_test)
    resultat = round(model.score(x_train, y_train)*100, 3)
    print('Accuracy : ',resultat)
    print()
    return resultat

def crossValScore_mean(model, cv):#____trg,adding, still testing
    n = cv
    m = model
    cvs = cross_val_score(m, x_train, y_train, n, scoring='accuracy')
    res = cvs.mean()
    print('crossVal moyenne:', res)
    return res
```

```python
x_train = train.drop('Survived', axis = 1)
y_train = train['Survived']

x_test = test
```

```python
models = [RandomForestClassifier(), LogisticRegression(), LinearRegression()]
for model in models :
    accuracy_model_bis(model, x_train, y_train, x_test)
    #crossValScore_mean(model, 5)
```

On peut remarquer selon le tableau de récapitulatif qui suit que les modèles utilisant des arbres sont beaucoup plus efficaces que les autres. En effet, en mesurant l' « Accuracy » de chaque modèle, on obtient :

```
RandomForestClassifier()
Accuracy :  89.787

LogisticRegression()
Accuracy :  79.012

LinearRegression()
Accuracy :  38.788
```

Puis, on voit aussi que l' « Accuracy » des modèles ne varient pas beaucoup après le passage de « Cross Validation ». On obtient le tableau suivant comme tableau de récapitulatif des résultats des modèles.

Cross validation

```
from sklearn.model_selection import cross_val_score
```

Modèle RandomForestClassifier

```
cross_RFC = cross_val_score(RandomForestClassifier(), x_train, y_train, cv=5, scoring='accuracy')
round(cross_RFC.mean(), 3)
```

```
0.804
```

Récapitulatif des résultats

Modèles	Accuracy	Moyenne CrossVal
RandomForestClassifier	89.787	0.8
LogisticRegression	79.012	0.787
LinearRegression	38.788	

Conclusion

On distingue clairement l'efficacité de l'algorithme de l'arbre de décision comparé aux autres modèles d'apprentissage automatique qu'on avait choisi avec un taux de 89.787.