Titanic : Machine Learning

Projet kaggle :

« Dans ce défi, nous vous demandons de compléter l'analyse des types de personnes susceptibles de survivre. En particulier, nous vous demandons d'appliquer les outils d'apprentissage par machine pour prédire quels passagers ont survécu à la tragédie. »

Source : https://www.kaggle.com/c/titanic

Les outils

- **Programmation en python**
- **IDE : Pycharm par jetbrains**
- **Pandas : librairie pour data analisis**
- **Seaborn et matplotlib.pyplot : librairie pour visualisation de données**
- **Sklearn : librairie l’algorithme de machine learning**

Importation des librairie

```
# librairie pour data analisis
import pandas as pd

# librairie pour visualisation de données
import seaborn as sns
import matplotlib.pyplot as plt

# Algo random forest
from sklearn.ensemble import RandomForestClassifier

# Importation des fichiers
train_df = pd.read_csv('train.csv')
test_df = pd.read_csv('test.csv')
combine = [train_df, test_df]
```

Traitement des données

```
for dataset in combine:
    # convertir String de Sex en integer
  dataset['Sex'] = dataset['Sex'].map({'female': 1, 'male': 0}).astype(int)
    # Répartition en tranche d'age
  dataset.loc[dataset['Age'].isnull(), 'Age'] = 0
  dataset.loc[dataset['Age'] <= 16, 'Age'] = 1
  dataset.loc[(dataset['Age'] > 16) & (dataset['Age'] <= 30), 'Age'] = 2
  dataset.loc[(dataset['Age'] > 30) & (dataset['Age'] <= 60), 'Age'] = 3
  dataset.loc[dataset['Age'] > 60, 'Age'] = 4
    # Convertir String de Embarked en integer
  dataset.loc[dataset['Embarked'].isnull(), 'Embarked'] = 0
  dataset.loc[dataset['Embarked'] == "C", 'Embarked'] = 1
  dataset.loc[dataset['Embarked'] == "Q", 'Embarked'] = 2
  dataset.loc[dataset['Embarked'] == "S", 'Embarked'] = 3
    # Répartition en tranche de prix
  dataset.loc[dataset['Fare'] < 7, 'Fare'] = 0
  dataset.loc[(dataset['Fare'] >= 7) & (dataset['Fare'] <= 14), 'Fare'] = 1
  dataset.loc[(dataset['Fare'] >= 14) & (dataset['Fare'] <= 31), 'Fare'] = 2
  dataset.loc[dataset['Fare'] > 31, 'Fare'] = 3
    # Creation d'un nouveau jeu de donnée : si la personne et seul ou non
  dataset['seul'] = 0
  dataset.loc[dataset['SibSp'] + dataset['Parch'] == 0, 'seul'] = 1
    # Suppretion des données non traité
train_df = train_df.drop(['Ticket', 'Cabin', 'Name'], axis=1)  # Supp
test_df = test_df.drop(['Ticket', 'Cabin',  'Name'], axis=1)  # Supp
```

Description de la table après traitement : train_df.describe()

	PassengerId	Survived	Pclass	Sex	Age \
count	891.000000	891.0000	891.0000	891.0000	891.000000
mean	446.000000	0.383838	2.308642	0.352413	2.056117
std	257.353842	0.486592	0.836071	0.477990	0.851432
min	1.000000	0.000000	1.000000	0.000000	1.000000
25%	223.500000	0.000000	2.000000	0.000000	1.000000
50%	446.000000	0.000000	3.000000	0.000000	2.000000
75%	668.500000	1.000000	3.000000	1.000000	3.000000
max	891.000000	1.000000	3.000000	1.000000	4.000000

	SibSp	Parch	Fare	seul
count	891.0000	891.0000	891.0000	891.000000
mean	0.523008	0.381594	1.727273	0.602694
std	1.102743	0.806057	0.871991	0.489615
min	0.000000	0.000000	0.000000	0.000000
25%	0.000000	0.000000	1.000000	0.000000
50%	0.000000	0.000000	2.000000	1.000000
75%	1.000000	0.000000	2.000000	1.000000
max	8.000000	6.000000	3.000000	1.000000

Description de la table après traitement

- Survivant en fonction des classes :
- ```
    Pclass  Survived
  0       1  0.629630
  1       2  0.472826
  2       3  0.242363
  ```
- ---
- Survivant en fonction de leur genre :
- ```
    Sex  Survived
  0    0  0.188908
  1    1  0.742038
  ```
- ---
- Survivant en fonction du prix :
- ```
    Fare  Survived
  0   0.0  0.071429
  1   1.0  0.266504
  2   2.0  0.439655
  3   3.0  0.581081
  ```
- ---
- Survivant en fonction de si la personne et seul ou non :
- ```
    seul  Survived
  0     0  0.505650
  1     1  0.303538
  ```

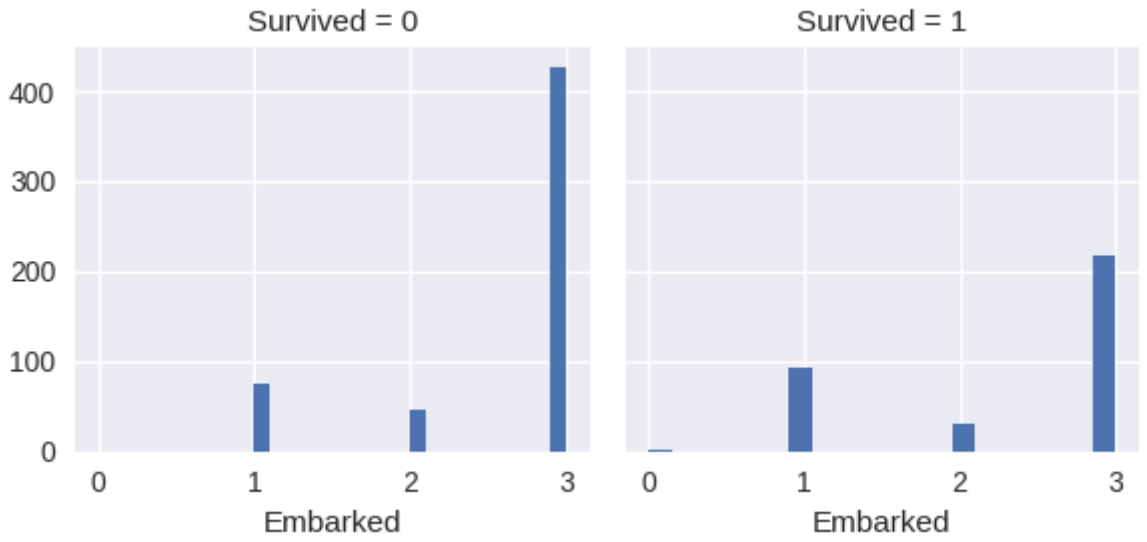
```
train_df[['Pclass', 'Survived']].groupby(['Pclass']
```

```
train_df[['Sex', 'Survived']].groupby(['Sex']
```

```
train_df[['Fare', 'Survived']].groupby(['Fare']
```

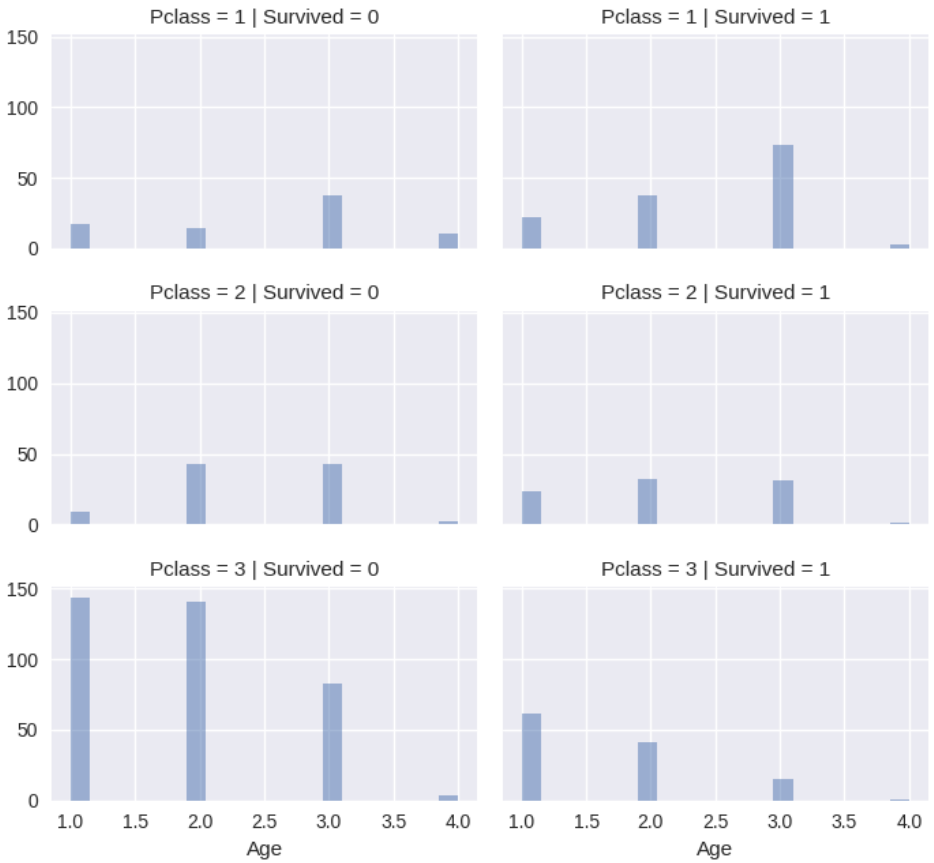
```
train_df[['seul', 'Survived']].groupby(['seul']
```

Visualisation de quelques données




```
g = sns.FacetGrid(train_df, col='Survived')
g.map(plt.hist, 'Embarked', bins=20)
```

Visualisation de quelques données




```
i = sns.FacetGrid(train_df,
col='Survived', row='Pclass', size=2.2,
aspect=1.6)

i.map(plt.hist, 'Age', alpha=.5,
bins=20)
```

Application de l’algorithme de random forest

```
X_train = train_df.drop("Survived", axis=1)
Y_train = train_df["Survived"]

random_forest = RandomForestClassifier(n_estimators=500,
oob_score=True)

random_forest.fit(X_train, Y_train)

round(random_forest.oob_score_ * 100, 2)
```

Score : 78.11 ± 1