Fonction Pandas

Pour inspection

```
df.info() # Description :
df.head(5) # Description :
df.shape # Description :
df.columns # Description :
df["Q1"].nunique() # Description :
df["Q1"].unique() # Description :
df.columns.nunique() # Description :
df[Score].value_counts() # Description :
```

Pour manipuler les colonnes

```
# select a column
df["sepal_length"]# select multiple columns and create a new dataframe X
X = df[["sepal_length", "sepal_width", "species"]]# select a column by column
number
df.iloc[:, [1,3,4]]# drop a column from dataframe X
X = X.drop("sepalL", axis=1)# save all columns to a list
df.columns.tolist()# Rename columns
df.rename(columns={"old colum1": "new column1", "old column2": "new
column2"})# sorting values by column "sepalW" in ascending order
df.sort_values(by = "sepal_width", ascending = True)# add new calculated
column
df['newcol'] = df["sepal_length"]*2# create a conditional calculated column
df['newcol'] = ["short" if i<3 else "long" for i in df["sepal_width"]]
```

Pour manipuler les lignes

```
# select rows 3 to 10
df.iloc[3:10,]# select rows 3 to 49 and columns 1 to 3
df.iloc[3:50, 1:4]# randomly select 10 rows
df.sample(10)# find rows with specific strings
df[df["species"].isin(["Iris-setosa"])]# conditional filtering
df[df.sepal_length >= 5]# filtering rows with multiple values e.g. 0.2,
0.3
df[df["petal_width"].isin([0.2, 0.3])]# multi-conditional filtering
df[(df.petal_length > 1) & (df.species=="Iris-setosa") |
(df.sepal_width < 3)]# drop rows
df.drop(df.index[1]) # 1 is row index to be deleted
```

Autre

```
# data grouped by column "species"
X = df.groupby("species")# return mean values of a column
("sepal_length" ) grouped by "species" column
df.groupby("spp")["sepal_length"].mean()# return mean values of ALL
columns grouped by "species" category
df.groupby("species").mean()# get counts in different categories
df.groupby("spp").nunique()
```

Données manquantes

```
print(df.isnull()) : afficher une matrix True/False des données
manquante.
print(df.isnull().sum()) : afficher un résumé des données manquantes
df.dropna(
axis=0, # Whether to drop rows or columns
how='any', # Whether to drop records if 'all' or 'any' records are
missing
thresh=None, # How many columns/rows must be missing to drop
subset=None, # Which rows/columns to consider
inplace=False # Whether to drop in place (i.e., without needing to re-
assign) )
```

Remplissage des données manquantes

```
df = df.fillna(0) :  remplir par des 0
df = df.fillna({'Name': 'Someone', 'Age': 25, 'Location': 'USA'})
print(df)
df['Age'] = df['Age'].fillna(df['Age'].mean())
```

Données dupliquées

```
print(df.duplicated()) : identifier la redondance
print(df.duplicated().sum()) : un résumé
df.drop_duplicates(
subset=None, # Which columns to consider
keep='first', # Which duplicate record to keep
inplace=False, # Whether to drop in place ignore_
index=False # Whether to relabel the index )
df = df.sort_values(by='Date Modified', ascending=False) df =
df.drop_duplicates(subset=['Name', 'Age'], keep='first') print(df)
```

Remplacer une donnée

```
data_replaced = combined_series.replace([-99, -100], np.nan)
```

Nettoyage des données String

```
pd.get_dummies(df)
```

Supprimer une donnée

```
bucket = [1, 25, 35, 60, 80, 100]
cut_data = pd.cut(data, bucket)
```