Groupe3
Alexandre BOULAT
Olivia MASSIKINI
David LOUSTAU-CARRERE

Analyse exploratoire

Nous travaillons sur un jeu de données de bières avec pour objectif de pouvoir calculer le degrés d'alcool (ABV) ainsi que le niveau d'amertume (IBU) des différentes bières.

Pour cela nous avons un jeu de données constitué de 73861 lignes ainsi que de 23 colonnes. L'objectif va être de remettre en forme le fichier CSV ainsi que de supprimer les colonnes trop peu renseignées.

Description des données
À l'aide de la méthode info() de pandas, on peut visualiser la description de chaque type de variables.

```
In [435]: df.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 73861 entries, 0 to 73860
Data columns (total 23 columns):
 #   Column         Non-Null Count  Dtype
---  ------         --------------  -----
 0   BeerID         73861 non-null  int64
 1   Name           73860 non-null  object
 2   URL            73861 non-null  object
 3   Style          73265 non-null  object
 4   StyleID        73861 non-null  int64
 5   Size(L)        73861 non-null  float64
 6   OG             73861 non-null  float64
 7   FG             73861 non-null  float64
 8   ABV            73861 non-null  float64
 9   IBU            73861 non-null  float64
 10  Color          73861 non-null  float64
 11  BoilSize       73861 non-null  float64
 12  BoilTime       73861 non-null  int64
 13  BoilGravity    70871 non-null  float64
 14  Efficiency     73861 non-null  float64
 15  MashThickness  43997 non-null  float64
 16  SugarScale     73861 non-null  object
 17  BrewMethod     73861 non-null  object
 18  PitchRate      34609 non-null  float64
 19  PrimaryTemp    51199 non-null  float64
 20  PrimingMethod  6766 non-null   object
 21  PrimingAmount  4774 non-null   object
 22  UserId         23371 non-null  float64
dtypes: float64(13), int64(3), object(7)
memory usage: 13.0+ MB
```

Cartographie de la distribution des valeurs manquantes
Pour sélectionner les données utiles ou inutiles nous avons mis en place une HeatMap des valeurs manquantes.

```
# Heatmap des valeurs manquantes
sns.heatmap(df.isnull(), cbar='False')
```

```
<Axes: >
```

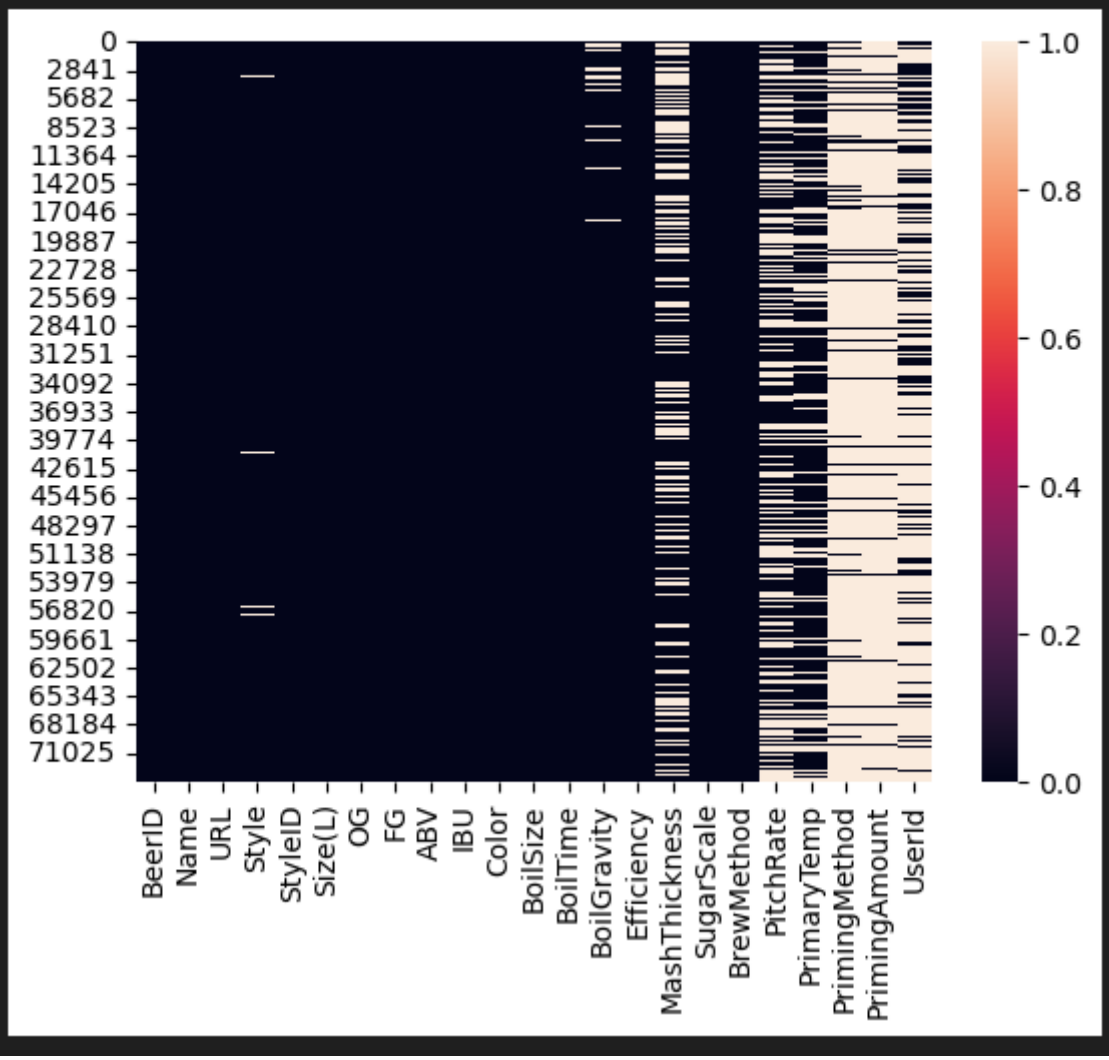
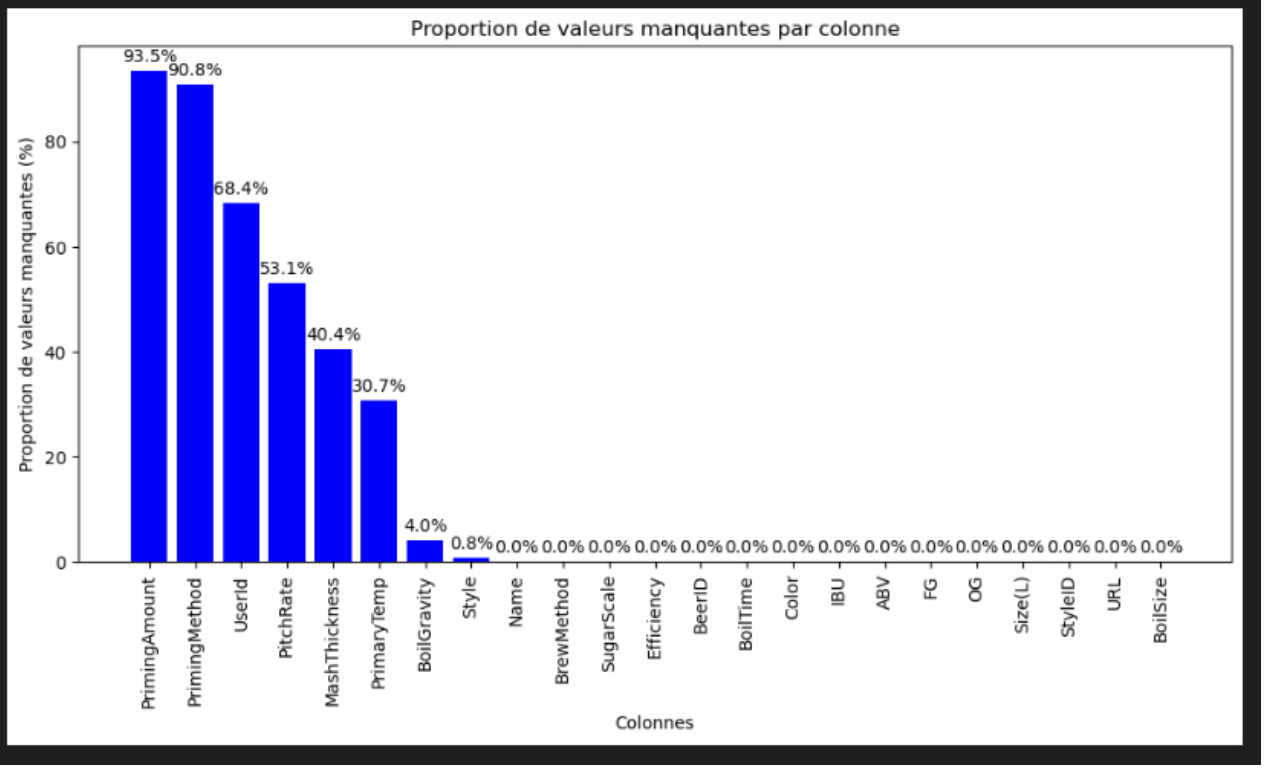


Diagramme en barre du pourcentage des valeurs manquantes

```python
# Tracer le graphique

plt.figure(figsize=(10, 6))

bars = plt.bar(missing_data.index, missing_data * 100, color='blue')
plt.bar(missing_data.index, missing_data * 100, color='blue')

plt.title('Proportion de valeurs manquantes par colonne')
plt.xlabel('Colonnes')
plt.ylabel('Proportion de valeurs manquantes (%)')
plt.xticks(rotation=90)

for bar, proportion in zip(bars, missing_data):
    plt.text(bar.get_x() + bar.get_width() / 2, bar.get_height() + 1, f'{proportion:.1%}', ha='center',

plt.tight_layout()
plt.show()
```



Nous pouvons voir que les colonnes MashThickness, PitchRate,PrimingMethod, PrimingAmount, UserId sont trop peu remplis pour être utilisées. Par contre nous conserverons la donnée PrimaryTemp que nous inputerons avec la valeur médiane du jeu de données.

Analyse statistique de quelques variables

```
In [436]: df.describe()
```

Out[436]:

	BeerID	StyleID	Size(L)	OG	FG	ABV	IBU	Color	BoilSize	BoilTime
count	73861.000000	73861.000000	73861.000000	73861.000000	73861.000000	73861.000000	73861.000000	73861.000000	73861.000000	73861.000000
mean	36931.000000	60.179432	43.929775	1.406266	1.075865	6.136865	44.276186	13.404989	49.724919	65.074870
std	21321.978453	56.811462	180.373492	2.196908	0.432524	1.883510	42.945508	11.944511	193.246427	15.024228
min	1.000000	1.000000	1.000000	1.000000	-0.003000	0.000000	0.000000	0.000000	1.000000	0.000000
25%	18466.000000	10.000000	18.930000	1.051000	1.011000	5.080000	23.370000	5.170000	20.820000	60.000000
50%	36931.000000	35.000000	20.820000	1.058000	1.013000	5.790000	35.770000	8.440000	27.440000	60.000000
75%	55396.000000	111.000000	23.660000	1.069000	1.017000	6.830000	56.380000	16.790000	30.000000	60.000000
max	73861.000000	176.000000	9200.000000	34.034500	23.424600	54.720000	3409.300000	186.000000	9700.000000	240.000000

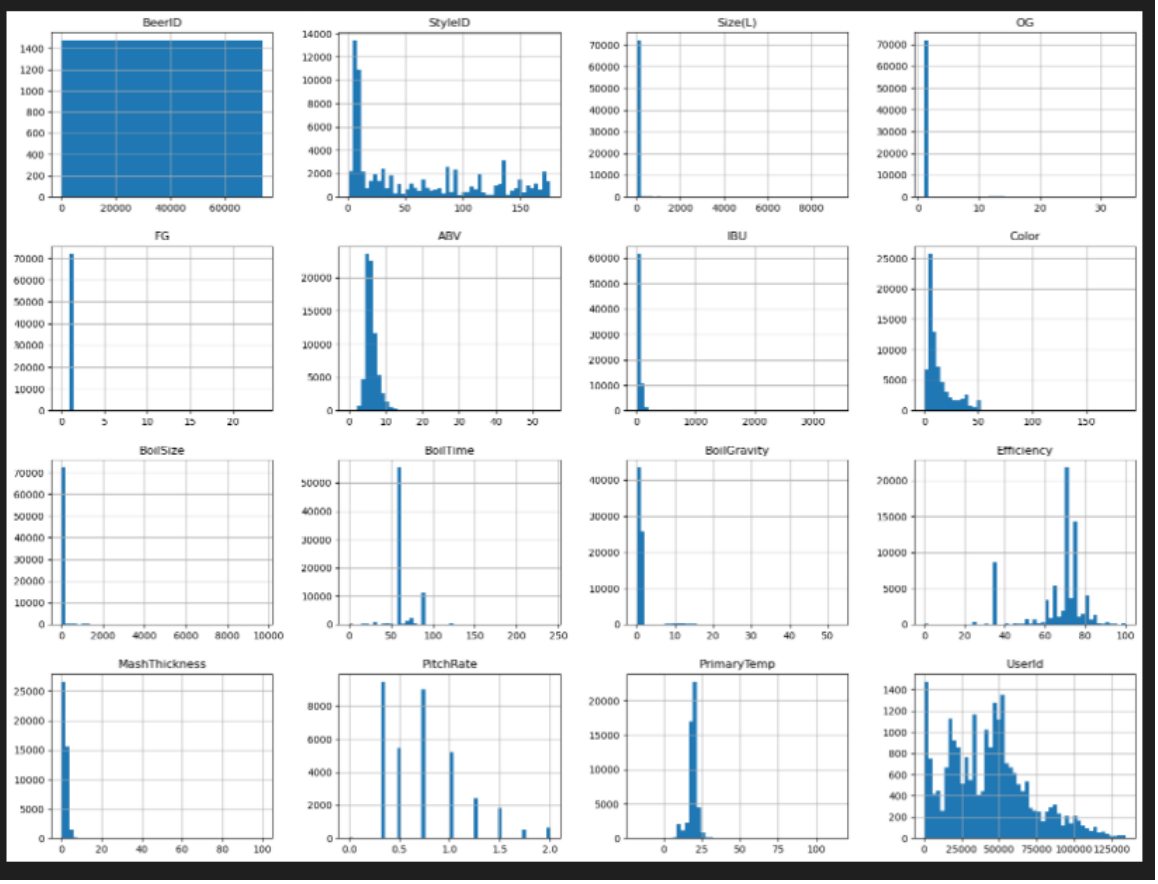
Type *Markdown* and LaTeX: α^2

A l'aide de la méthode describe() de pandas, nous pouvons visualiser quelques paramètres statistiques de notre jeu de données dont, la moyenne, l'écart-type, le premier quartile(25%), le troisième quartile(75%), la médiane, le minimun et le maximun.

Analyse descriptive : Histogramme de la distribution des variables

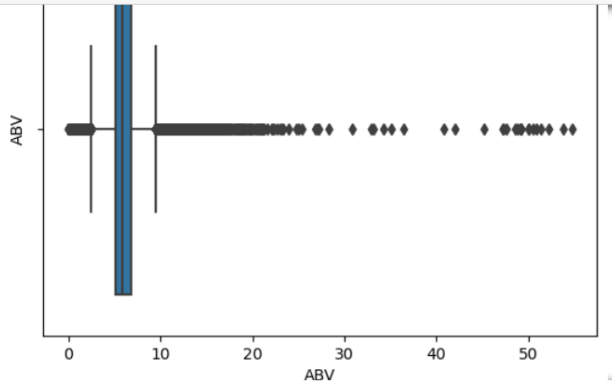
Visualisation de la distribution des données à l'aide d'histogramme.
À l aide de la méthode hist() de pandas, nous pouvons visualiser la distribution des variables numériques

```
# Visualisations sous la forme d'histogrammes

df_num.hist(bins=50, figsize=(20, 15))

plt.show()
```



Analyse descriptive : **la boîte à moustache de la distribution d' une de nos variables**

```
In [442]: # Sous la forme de boîtes à moustaches

          plt.figure(figsize=(20, 15))

          for col in df_num.columns:
              plt.figure()
              sns.boxplot(data=df_num, x=col, orient='h')
              plt.title(f'Diagramme en boîtes de {col}')
              plt.ylabel(col)
              plt.show()
```



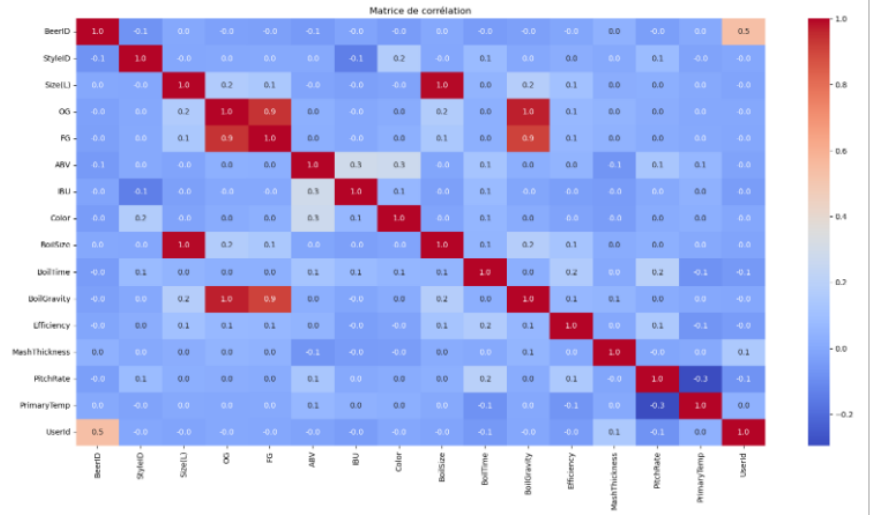
Analyse statistique : étude de corrélation

On remarque que BoilSize est complètement corrélé au Volume du Brassin, nous créerons une nouvelle variable à partir de ces deux là : Le volume d'eau en L utilisé pour fabriquer un litre de bière.

BoilGravity est quasiment complétement correlé à FG et OG, nous ne la conserverons donc pas dans nos analyses.

FG et OG sont complètement corrélées, nous créerons une variable à partir de ces deux là dans la partie Feature Engineering qui sera FG - OG correspondant à la différence de densité entre le moût après fermentation et avant fermentation, en unités de densité.

```
In [443]: # Etude des correlations entre les variables de notre jeu de données :

          # Calculer la matrice de corrélation avec les variables quantitatives uniquement
          correlation_matrix = df_num.corr()

          # Afficher la matrice de corrélation avec une heatmap
          plt.figure(figsize=(20, 10))
          sns.heatmap(correlation_matrix, annot=True, fmt=".1f", cmap= "coolwarm")

          #affichage de la matrice de corrélation
          plt.title("Matrice de corrélation")
          plt.show()
```



Analyse statistique: Test ANOVA

Test ANOVA entre nos variables cibles et nos variables catégorielles

In [453]:
```
targets = ['ABV', 'IBU']

# Variables catégorielles
liste_cat = ['Style','SugarScale', 'BrewMethod']

for target in targets:
    print(f"Analyse de variance pour la variable cible: {target}\n")

    for cat in liste_cat:
        print(f"Variable catégorielle: {cat}")

        # Séparer les groupes en fonction de la variable catégorielle
        groupes = []
        for group_name, data in df.groupby(df[cat]):
            groupes.append(data[target])

        # Effectuer le test ANOVA
        f_statistic, p_value = stats.f_oneway(*groupes)


        print(f"Statistique F: {f_statistic:.2f}")
        print(f"Valeur p: {p_value:.4f}")

        # Interpréter le résultat

        if p_value < 0.05:
            print("Rejet de l'hypothèse nulle : Il y a des différences significatives entre les groupes.")
        else:
            print("Échec de rejeter l'hypothèse nulle : Aucune différence significative entre les groupes.")
        print("----------------------------------------")
    print("\n========================================\n")
```

Résultat du Test ANOVA

```
Analyse de variance pour la variable cible: ABV

Variable catégorielle: Style
Statistique F: 299.68
Valeur p: 0.0000
Rejet de l'hypothèse nulle : Il y a des différences significatives entre les groupes.
----------------------------------------
Variable catégorielle: SugarScale
Statistique F: 10.21
Valeur p: 0.0014
Rejet de l'hypothèse nulle : Il y a des différences significatives entre les groupes.
----------------------------------------
Variable catégorielle: BrewMethod
Statistique F: 25.73
Valeur p: 0.0000
Rejet de l'hypothèse nulle : Il y a des différences significatives entre les groupes.
----------------------------------------

========================================

Analyse de variance pour la variable cible: IBU

Variable catégorielle: Style
Statistique F: 171.67
Valeur p: 0.0000
Rejet de l'hypothèse nulle : Il y a des différences significatives entre les groupes.
----------------------------------------
Variable catégorielle: SugarScale
Statistique F: 16.59
Valeur p: 0.0000
Rejet de l'hypothèse nulle : Il y a des différences significatives entre les groupes.
----------------------------------------
Variable catégorielle: BrewMethod
Statistique F: 1.98
Valeur p: 0.1141
Échec de rejeter l'hypothèse nulle : Aucune différence significative entre les groupes.
----------------------------------------

========================================
```

Ces résultats sont à relativiser car il faut vérifier si les distributions et les variances des données dans chaque groupe sont à peu près similaires.

La méthode de brassage ne semble pas influer sur l'Amertume d'une bière.

Nettoyage du jeu de données, conformément à nos analyses

Suppression des variables trop peu renseignées (seuil à 40% de valeurs manquantes)

```
5]: df = df.drop(['PrimingAmount', 'PrimingMethod','UserId', 'PitchRate', 'MashThickness'], axis =1)
```

Suppression des variables qui n'ont pas de sens pour notre problématique

```
']: df = df.drop(['BeerID', 'Name', 'URL', 'StyleID'], axis = 1)
```

Suppression de la variable BoilGravity, qui a une corrélation linéaire quasi-parfaite avec OG et FG

```
59]: df = df.drop(['BoilGravity'], axis = 1)
```

Suppression des individus contenant des valeurs aberrantes par l'application de filtres

```
8]: # Une temperature de début de fermentation ne peut pas être négative (on conserve les NaN pour imputer plus tard)
    df = df.loc[(df['PrimaryTemp'].isnull()) | (df['PrimaryTemp'] > 0)]
```

```
9]: # La densité du moût après fermentation ne peut pas être négative
    df = df[df['FG'] > 0]
```

Feature Engineering

Feature Engineering conforme aux analyses exploratoires

Nous souhaitons effectuer une classification concernant l'amertume des bières, conformément aux exigences du milieu :

IBU < 20 : peu d'amertume - 0

IBU 20 à 40 : amertume modérée - 1

IBU 40 à 60 : amertume prononcée - 2

IBU 60 à 80 : amertume intense - 3

IBU > 80 : amertume très intense - 4

Source : https://unepetitemousse.fr/blog/biere-amere/

Pour la suite, nous créeons des variables à partir de deux variables fortement corrélées linéairement et qui font plus sens que les variables initiales compte tenu de notre problématique

```
# Création d'une variable correspondant à la différence de densité du moût entre le début et la fin de la fermentation
df['diff_densite_fermentation'] = df['FG'] - df['OG']
```

```
# Suppression des variables 'FG' et 'OG'
df = df.drop(['FG', 'OG'], axis = 1)
```

```
# Création d'une variable correspondant à la quantité d'eau utilisée pour l'ébulition du moût par litre de bière brassée
df['qte_eau_litre_biere'] = df['BoilSize'] / df['Size(L)']
```

```
# Suppression des variables 'BoilSize' et 'Size(L)'
df = df.drop(['BoilSize', 'Size(L)'], axis = 1)
```

```
# Application d'un filtre pour retirer les bières qui sont brassées avec moins de 0.33L d'eau/L_bière
df = df[df['qte_eau_litre_biere'] > 1/3]
```

```
df[df['qte_eau_litre_biere'] >= 3].shape
```

```
(85, 11)
```

```
# Suppression des valeurs extrêmes pour la variable 'qte_eau_litre_biere' (on la souhaite inférieure à 3 strictement)
df = df[df['qte_eau_litre_biere'] < 3]
```

```
df[df['diff_densite_fermentation'] <= -17].shape
```

```
(71, 11)
```

```
# Suppression des valeurs extrêmes pour la variable 'diff_densite_fermentation' (on souhaite retirer un millième du jeu)
df = df[df['diff_densite_fermentation'] >= -17]
```

Résultat final de notre jeu de données après traitement
La variable PrimaryTemp sera imputé par la médiane après le split d'entrainement du jeu de données.

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 72416 entries, 0 to 72896
Data columns (total 11 columns):
 #   Column                      Non-Null Count  Dtype  
---  ------                      --------------  -----  
 0   Style                       72416 non-null  object 
 1   ABV                         72416 non-null  float64
 2   Color                       72416 non-null  float64
 3   BoilTime                    72416 non-null  int64  
 4   Efficiency                  72416 non-null  float64
 5   SugarScale                  72416 non-null  object 
 6   BrewMethod                  72416 non-null  object 
 7   PrimaryTemp                 50351 non-null  float64
 8   IBU_LABEL                   72416 non-null  int32  
 9   diff_densite_fermentation   72416 non-null  float64
 10  qte_eau_litre_biere         72416 non-null  float64
dtypes: float64(6), int32(1), int64(1), object(3)
memory usage: 6.4+ MB
```