МІНІСТЕРСТВО ОСВІТИ І НАУКИ, МОЛОДІ ТА СПОРТУ УКРАЇНИ
НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ ІМЕНІ ІГОРЯ СІКОРСЬКОГО»
ФІЗИКО-ТЕХНІЧНИЙ ІНСТИТУТ
Кафедра Інформаційної Безпеки

Засоби підготовки та аналізу даних

Лабораторна робота №3

Структури для роботи з великими обсягами даних в Python

Мета: отримати навички роботи із структурами для зберігання в Python (`python`, `numpy`, `pandas`, `numpy array`, `dataframe`, `timeit`)

Основні поняття: `numpy` масиви, кортежі, списки, фрейми, профілювання.

Перевірив:

_____

Виконав:
студент II курсу
групи ФБ-01
Сахній Н.Р.

Київ 2022

Другий рівень (ускладнений)

Першим кроком є вибір датасету із архіву https://archive.ics.uci.edu/ml/index.php/
Датасет має відповідати таким вимогам:

- Data Set Characteristics: Multivariate
- Attribute Characteristics: Categorical, Integer, Real
- Number of Attributes: at least 2 integers/real
- Missing Values? YES!!!!!

Отже, виберемо **Cylinder Bands Data Set**
https://archive.ics.uci.edu/ml/datasets/Cylinder+Bands

Cylinder Bands Data Set

Download: Data Folder, Data Set Description

Abstract: Used in decision tree induction for mitigating process delays known as "cylinder bands" in rotogravure printing

Data Set Characteristics:	Multivariate	**Number of Instances:**	512	**Area:**	Physical
Attribute Characteristics:	Categorical, Integer, Real	**Number of Attributes:**	39	**Date Donated**	1995-08-01
Associated Tasks:	Classification	**Missing Values?**	Yes	**Number of Web Hits:**	93444

bands.names: Блокнот

Файл Редагування Формат Вигляд Довідка

```
Attribute Information:
        1. Timestamp: numeric;19500101 - 21001231
        2. Cylinder number: nominal
        3. Customer: nominal;
        4. Job number: nominal;
        5. Grain screened: nominal; yes, no
        6. Ink color: nominal;  key, type
        7. Proof on ctd ink:  nominal;  yes, no
        8. Blade mfg: nominal;  benton, daetwyler, uddeholm
        9. Cylinder division: nominal;  gallatin, warsaw, mattoon
       10. Paper type: nominal;  uncoated, coated, super
       11. Ink type: nominal;  uncoated, coated, cover
       12. Direct steam: nominal; use; yes, no *
       13. Solvent type: nominal;  xylol, lactol, naptha, line, other
       14. Type on cylinder:  nominal;  yes, no
       15. Press type: nominal; use; 70 wood hoe, 70 motter, 70 albert, 94 motter
       16. Press: nominal;  821, 802, 813, 824, 815, 816, 827, 828
       17. Unit number: nominal;  1, 2, 3, 4, 5, 6, 7, 8, 9, 10
       18. Cylinder size: nominal;  catalog, spiegel, tabloid
       19. Paper mill location: nominal; north us, south us, canadian,
                          scandanavian, mid european
       20. Plating tank: nominal; 1910, 1911, other
       21. Proof cut: numeric;  0-100
       22. Viscosity: numeric;  0-100
       23. Caliper: numeric;  0-1.0
       24. Ink temperature: numeric;  5-30
       25. Humifity: numeric;  5-120
       26. Roughness: numeric;  0-2
       27. Blade pressure: numeric;  10-75
       28. Varnish pct: numeric;  0-100
       29. Press speed: numeric;  0-4000
       30. Ink pct: numeric;  0-100
       31. Solvent pct: numeric;  0-100
       32. ESA Voltage: numeric;  0-16
       33. ESA Amperage: numeric;  0-10
       34. Wax: numeric ;  0-4.0
       35. Hardener:  numeric; 0-3.0
       36. Roller durometer:  numeric;  15-120
       37. Current density:  numeric;  20-50
       38. Anode space ratio:  numeric;  70-130
       39. Chrome content: numeric; 80-120
       40. Band type: nominal; class; band, no band *
```

Опис атрибутів датасету

Завдання другого рівня

Виконати всі завдання, використовуючи як `numpy array`, так і `dataframe`

1. Поборотися із зниклими даними. Для цього в допомогу вам Медіум (https://towardsdatascience.com/how-to-handle-missing-data-8646b18db0d4).

1. Поборемося із зниклими даними ↓

In [85]:
```
from urllib.request import urlopen
import pandas as pd
import numpy as np


url = 'https://archive.ics.uci.edu/ml/machine-learning-databases/cylinder-bands/bands.data'

# Відкриття WEB-сторінки можна зробити наступним чином:
webpage = urlopen(url)
text = webpage.read()

# Створити новий файл за допомоги функції open
out = open('bands.data', 'wb')
# Після відкриття у змінній text міститься текст із WEB-сторінки, який тепер можна записати у файл
out.write(text)
out.close()


# Визначення заголовків для стовців датафрейму
headers = ['Timestamp', 'Cylinder number', 'Customer', 'Job number', 'Grain screened', 'Ink color', 'Proof on ink',
           'Blade mfg', 'Cylinder division', 'Paper type', 'Ink type', 'Direct steam', 'Solvent type',
           'Type on cylinder', 'Press type', 'Press', 'Unit number', 'Cylinder size', 'Paper mill location',
           'Plating tank', 'Proof cut', 'Viscosity', 'Caliper', 'Ink temperature', 'Humifity', 'Roughness',
           'Blade pressure', 'Varnish pct', 'Press speed', 'Ink pct', 'Solvent pct', 'ESA Voltage', 'ESA Amperage',
           'Wax', 'Hardener', 'Roller durometer', 'Current density', 'Anode space ratio', 'Chrome content', 'Band type']
# Відкриття файлу та його запис у датафрейм
df = pd.read_csv('./bands.data', sep=',', header=1, names=headers)

# Відкинемо всі None (Порожні значення). При цьому враховуємо, що None у файлі позначаються як '?'
df.replace('?', np.nan, inplace=True)
df.dropna(inplace=True)

df
```

Out[85]:

	Timestamp	Cylinder number	Customer	Job number	Grain screened	Ink color	Proof on ink	Blade mfg	Cylinder division	Paper type	...	Solvent pct	ESA Voltage	ESA Amperage	Wax	Hardener	Ro durom
1	19910104	T133	MASSEY	39039	YES	KEY	YES	BENTON	GALLATIN	UNCOATED	...	38.8	0	0	2.5	1.3	
3	19910104	T218	MASSEY	38039	YES	KEY	YES	BENTON	GALLATIN	UNCOATED	...	37.6	5	0	2.5	0.8	
4	19910111	X249	ROSES	35751	NO	KEY	YES	BENTON	GALLATIN	COATED	...	37.5	6	0	2.5	0.6	
5	19910111	X788	ROSES	35751	NO	KEY	YES	BENTON	GALLATIN	COATED	...	37.5	6	0	2.5	1.1	
6	19910112	M372	MODMAT	47201	YES	KEY	YES	BENTON	GALLATIN	UNCOATED	...	39.8	1.5	0	3	1	
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	
422	19901211	X242	AMES	34590	NO	KEY	YES	BENTON	GALLATIN	COATED	...	41.2	8	0	3	1	
424	19901214	X108	ECKERDS	34693	NO	KEY	YES	BENTON	GALLATIN	COATED	...	37.5	1	0	2.5	1.5	
425	19901218	X80	ECKERDS	34694	NO	KEY	YES	BENTON	GALLATIN	COATED	...	39.5	4.5	0	1.9	1.3	
426	19901218	F482	DOWNS	35525	YES	KEY	YES	BENTON	GALLATIN	UNCOATED	...	36.1	4	0	3	1	
427	19901230	X388	TVGUIDE	25502	YES	KEY	YES	BENTON	GALLATIN	UNCOATED	...	53.4	0	0	3	0.9	

276 rows × 40 columns

2. Пронормувати вибраний датасет або стандартизувати його (нормалізація і стандартизація мають бути реалізовані як окремі функції без застосування додаткових бібліотек, як наприклад sklearn.preprocessing).

In [86]:
```
float_columns = ['Proof cut', 'Viscosity', 'Caliper', 'Ink temperature', 'Humifity', 'Roughness',
                 'Blade pressure', 'Varnish pct', 'Press speed', 'Ink pct', 'Solvent pct', 'ESA Voltage', 'ESA Amperage',
                 'Wax', 'Hardener', 'Roller durometer', 'Current density', 'Anode space ratio', 'Chrome content']
```

```
df_copy = df.copy()
for header in float_columns:
    # Змінимо тип даних потрібних нам стовпців на числовий
    df_copy[header] = pd.to_numeric(df_copy[header], errors='coerce')

norm_data = df_copy.copy()
for column in float_columns:
    # Пронормуємо відповідні номерні значення у датафреймі
    norm_data[column] = (norm_data[column] - (norm_data[column].min())) / (norm_data[column].max() - (norm_data[column].min()))

norm_data.dropna(axis=1)
```

Out[86]:

	Timestamp	Cylinder number	Customer	Job number	Grain screened	Ink color	Proof on ink	Blade mfg	Cylinder division	Paper type	...	Solvent pct	ESA Voltage	ESA Amperage	Wax	Hardener
1	19910104	T133	MASSEY	39039	YES	KEY	YES	BENTON	GALLATIN	UNCOATED	...	0.527508	0.00000	0.0	0.806452	0.433333
3	19910104	T218	MASSEY	38039	YES	KEY	YES	BENTON	GALLATIN	UNCOATED	...	0.488673	0.31250	0.0	0.806452	0.266667
4	19910111	X249	ROSES	35751	NO	KEY	YES	BENTON	GALLATIN	COATED	...	0.485437	0.37500	0.0	0.806452	0.200000
5	19910111	X788	ROSES	35751	NO	KEY	YES	BENTON	GALLATIN	COATED	...	0.485437	0.37500	0.0	0.806452	0.366667
6	19910112	M372	MODMAT	47201	YES	KEY	YES	BENTON	GALLATIN	UNCOATED	...	0.559871	0.09375	0.0	0.967742	0.333333
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
422	19901211	X242	AMES	34590	NO	KEY	YES	BENTON	GALLATIN	COATED	...	0.605178	0.50000	0.0	0.967742	0.333333
424	19901214	X108	ECKERDS	34693	NO	KEY	YES	BENTON	GALLATIN	COATED	...	0.485437	0.06250	0.0	0.806452	0.500000
425	19901218	X80	ECKERDS	34694	NO	KEY	YES	BENTON	GALLATIN	COATED	...	0.550162	0.28125	0.0	0.612903	0.433333
426	19901218	F482	DOWNS	35525	YES	KEY	YES	BENTON	GALLATIN	UNCOATED	...	0.440129	0.25000	0.0	0.967742	0.333333
427	19901230	X388	TVGUIDE	25502	YES	KEY	YES	BENTON	GALLATIN	UNCOATED	...	1.000000	0.00000	0.0	0.967742	0.300000

276 rows × 40 columns

2.2. Стандартизуємо дані ↓

In [87]:
```
standart_data = df_copy.copy()
for column in float_columns:
    standart_data[column] = (standart_data[column] - standart_data[column].mean()) / standart_data[column].std()

standart_data.dropna(axis=1)
```

Out[87]:

	Timestamp	Cylinder number	Customer	Job number	Grain screened	Ink color	Proof on ink	Blade mfg	Cylinder division	Paper type	...	Solvent pct	ESA Voltage	ESA Amperage	Wax	Harder
1	19910104	T133	MASSEY	39039	YES	KEY	YES	BENTON	GALLATIN	UNCOATED	...	0.136885	-0.558834	-0.065003	0.140443	1.0160
3	19910104	T218	MASSEY	38039	YES	KEY	YES	BENTON	GALLATIN	UNCOATED	...	-0.191972	1.289437	-0.065003	0.140443	-0.5163
4	19910111	X249	ROSES	35751	NO	KEY	YES	BENTON	GALLATIN	COATED	...	-0.219377	1.659092	-0.065003	0.140443	-1.1293
5	19910111	X788	ROSES	35751	NO	KEY	YES	BENTON	GALLATIN	COATED	...	-0.219377	1.659092	-0.065003	0.140443	0.4030
6	19910112	M372	MODMAT	47201	YES	KEY	YES	BENTON	GALLATIN	UNCOATED	...	0.410933	-0.004353	-0.065003	1.056810	0.0966
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	
422	19901211	X242	AMES	34590	NO	KEY	YES	BENTON	GALLATIN	COATED	...	0.794599	2.398400	-0.065003	1.056810	0.0966
424	19901214	X108	ECKERDS	34693	NO	KEY	YES	BENTON	GALLATIN	COATED	...	-0.219377	-0.189180	-0.065003	0.140443	1.6290
425	19901218	X80	ECKERDS	34694	NO	KEY	YES	BENTON	GALLATIN	COATED	...	0.328718	1.104610	-0.065003	-0.959197	1.0160
426	19901218	F482	DOWNS	35525	YES	KEY	YES	BENTON	GALLATIN	UNCOATED	...	-0.603044	0.919783	-0.065003	1.056810	0.0966
427	19901230	X388	TVGUIDE	25502	YES	KEY	YES	BENTON	GALLATIN	UNCOATED	...	4.137981	-0.558834	-0.065003	1.056810	-0.2098

276 rows × 40 columns

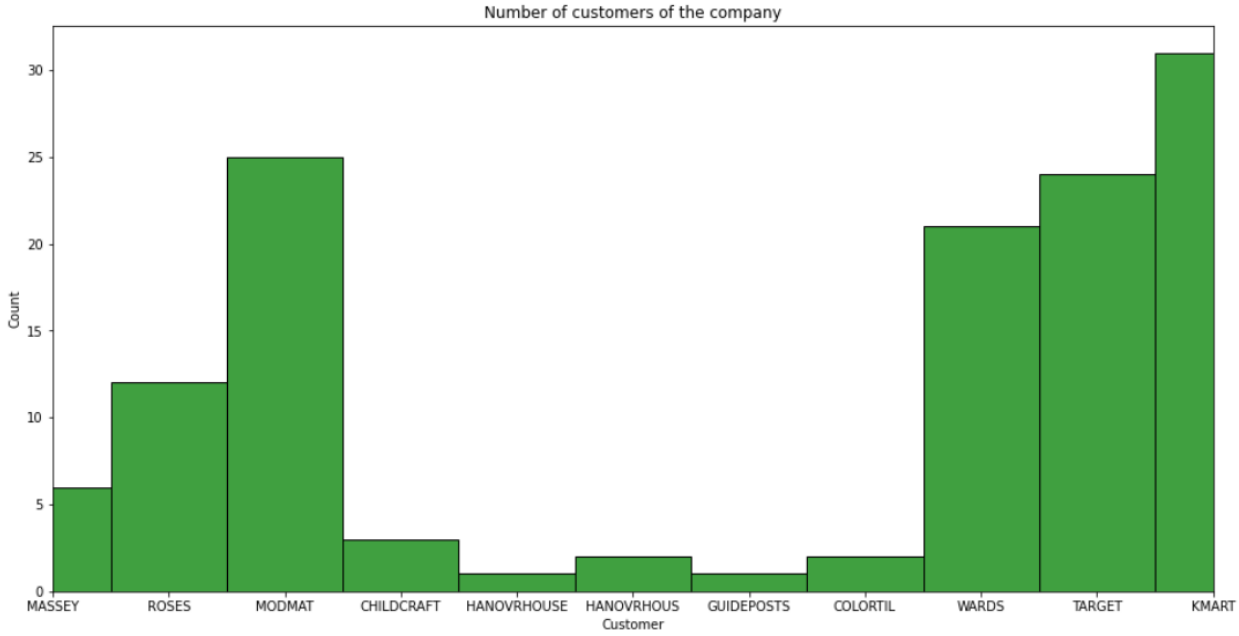
3. Збудувати гістограму по одному із атрибутів, що буде показувати на кількість елементів, що знаходяться у 10 діапазонах, які ви задасте.

3. Збудуємо гістограму по одному із атрибутів, що буде показувати на кількість елементів, що знаходяться у 10 діапазонах, які ми задамо ↓

In [88]:
```
import matplotlib.pyplot as plt
import seaborn as sns

plt.figure(figsize=(16, 8))
plt.title("Number of customers of the company")
```

```
# Задамо діапазон із десяти різних замовників
plt.xlim(0, 10)
# Значення stat='count' вказує, що зробити підрахунок елементів
sns.histplot(df['Customer'], stat='count', color='green')
```

Out[88]: <AxesSubplot:title={'center':'Number of customers of the company'}, xlabel='Customer', ylabel='Count'>

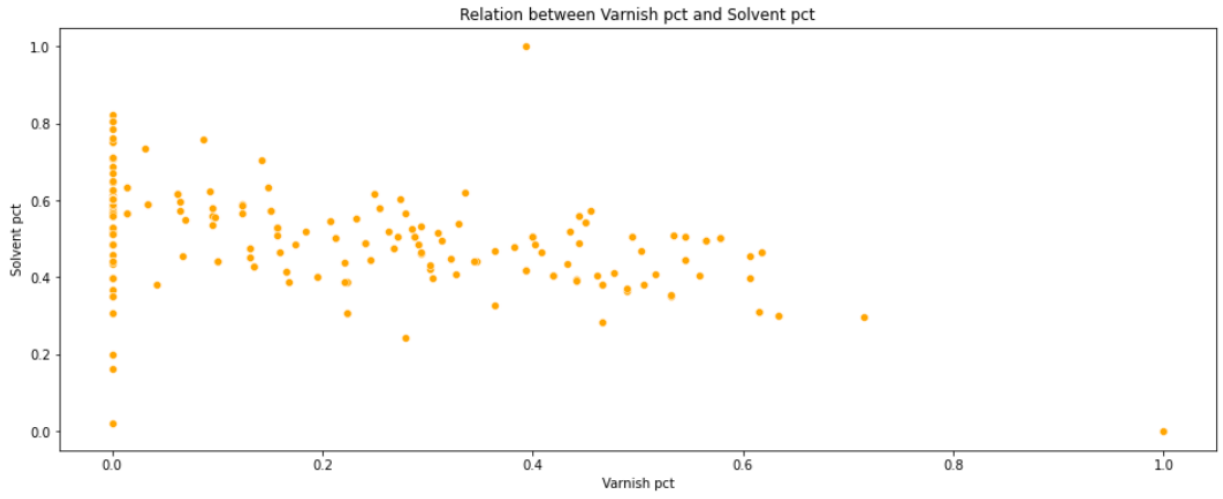


4. Збудувати графік залежності одного integer/real атрибута від іншого.

4. Побудуємо графік залежності одного integer/real атрибута від іншого ↓

In [89]:
```
plt.figure(figsize=(16, 6))
plt.title("Relation between Varnish pct and Solvent pct")

# Щоб показати залежність атрибутів, будемо використовувати пронормовані елементи
sns.scatterplot(x=norm_data['Varnish pct'], y=norm_data['Solvent pct'], color = 'orange')
```

Out[89]: <AxesSubplot:title={'center':'Relation between Varnish pct and Solvent pct'}, xlabel='Varnish pct', ylabel='Solvent pct'>



5. Підрахувати коефіцієнт Пірсона та Спірсона для двох integer/real атрибутів.

5. Підрахуємо коефіцієнт Пірсона та Спірсона для двох integer/real атрибутів ↓

In [90]:
```
from scipy import stats
x = df['Viscosity'].astype('float')
y = df['Solvent pct'].astype('float')

pearson = stats.pearsonr(x, y)
print(f"PearsonResult = {pearson} \n")

spearman = stats.spearmanr(x, y)
print(spearman)
```

```
PearsonResult = (-0.0005117874538250466, 0.9932468939778661)

SpearmanrResult(correlation=0.008626260535667703, pvalue=0.8865567731077645)
```

6. Провести One Hot Encoding категоріального string атрибуту.

In [91]:
```
from sklearn.preprocessing import LabelEncoder
from sklearn.preprocessing import OneHotEncoder


data = df['Paper mill location'].str.upper()
values = np.array(data)
print("Розташування паперової фабрики: \n", np.unique(values))

# Label Encoder
label_encoder = LabelEncoder()
integer_encoded = label_encoder.fit_transform(values)
print("\nInteger Encoding: \n", integer_encoded)


# Reshaping for OneHotEncoder
integer_encoded_reshape = integer_encoded.reshape(len(integer_encoded), 1)

# One Hot Encoder
one_hot_encoder = OneHotEncoder(sparse=False)
one_hot_encoded = one_hot_encoder.fit_transform(integer_encoded_reshape)
print("\nOne Hot Encoding: \n", one_hot_encoded)
```

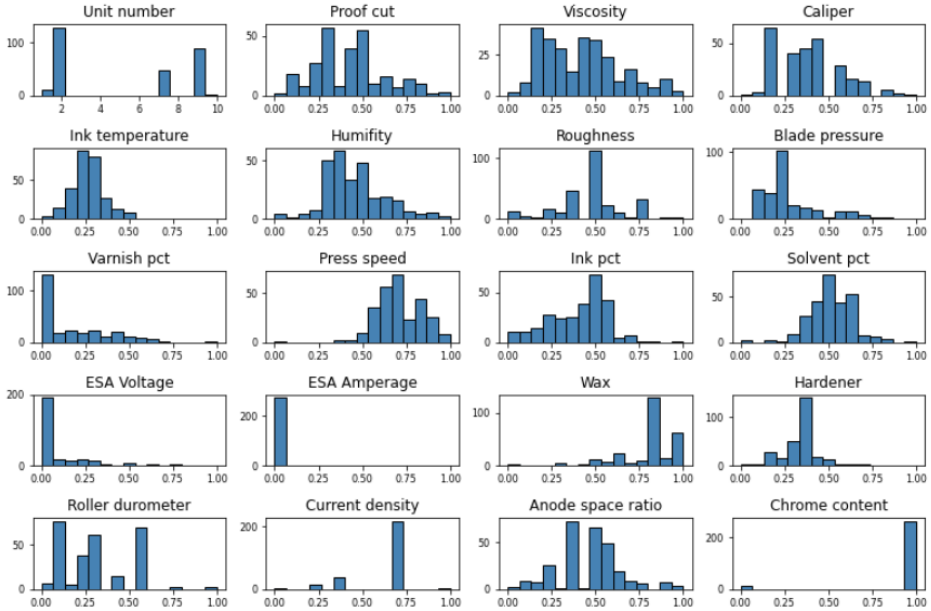
```
Розташування паперової фабрики:
 ['CANADIAN' 'MIDEUROPEAN' 'NORTHUS' 'SCANDANAVIAN' 'SOUTHUS']

Integer Encoding:
 [2 2 0 0 2 2 2 2 2 2 2 0 0 0 0 3 3 3 3 0 2 2 2 2 2 0 0 2 2 0 0 3 2 2 0 0 0
 2 2 2 4 4 4 4 0 2 2 0 2 3 0 2 0 2 2 2 0 0 0 0 0 0 0 0 0 2 0 0 2 0 0 2 0 2 2 2
 2 0 0 0 0 2 2 2 2 0 0 3 3 0 0 0 0 0 0 0 2 0 3 0 0 0 0 0 0 0 2 2 0 2 2 2 2 0 0
 2 2 2 0 0 0 2 2 2 0 2 2 2 0 0 0 0 2 2 2 2 0 0 2 2 0 0 0 2 2 0 0 0 0 0 0 0 0 2
 2 2 2 2 0 2 2 2 2 2 0 0 0 2 2 2 2 2 2 0 0 0 0 2 2 2 2 2 2 2 2 1 1 0 0 0 0 0 2
 2 0 0 0 2 2 2 2 2 2 2 0 2 2 2 1 0 0 1 1 2 2 1 1 1 0 2 2 2 2 2 0 0 0 2 2 2 2 0
 2 2 0 2 2 0 2 0 2 2 2 2 2 2 0 0 0 0 2 2 0 0 0 2 2 2 2 2 2 2 1 1 1 0 2 2 2 2 0
 2 2 0 0 0 2 2 2 2 2 0 2 0 0 0 0 0 2 2]

One Hot Encoding:
 [[0. 0. 1. 0. 0.]
 [0. 0. 1. 0. 0.]
 [1. 0. 0. 0. 0.]
 ...
 [1. 0. 0. 0. 0.]
 [0. 0. 1. 0. 0.]
 [0. 0. 1. 0. 0.]]
```

7. Провести візуалізацію багатовимірних даних, використовуючи приклади, наведені у медіумі: https://towardsdatascience.com/the-art-of-effective-visualization-of-multi-dimensional-data-6c7202990c57

7. Проведемо візуалізацію багатовимірних даних ↓

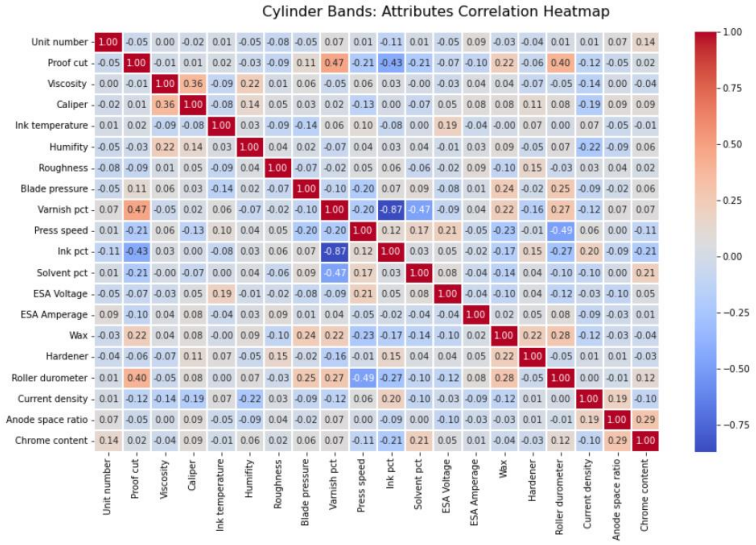
Visualizing data in One Dimension (1D)

In [92]:
```
# Several histograms from dataframe attributes

norm_data.hist(bins=15, color='steelblue', edgecolor='black', linewidth=1.0,
       xlabelsize=8, ylabelsize=8, grid=False)
plt.tight_layout(rect=(0, 0, 1.8, 1.8))
```



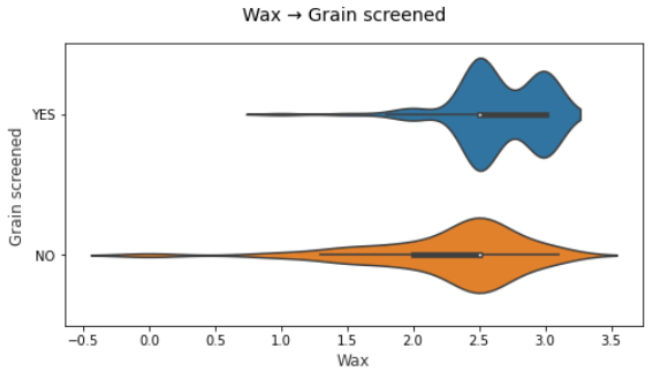
Visualizing data in One Dimension (2D)

In [93]:
```
# Visualizing data in One Dimension (2D)
# Correlation Matrix Heatmap

fm, ax = plt.subplots(figsize=(14, 8))
corr = norm_data.corr()
hm = sns.heatmap(round(corr, 2), annot=True, ax=ax, cmap='coolwarm', fmt='.2f',
                 linewidths=.05)
fm.subplots_adjust(top=0.93)
title = fm.suptitle('Cylinder Bands: Attributes Correlation Heatmap', fontsize=16)
```



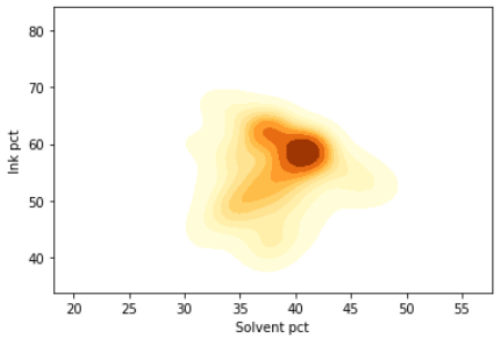
In [94]:
```
# Violin Plots

f, (ax) = plt.subplots(1, 1, figsize=(8, 4))
f.suptitle('Wax → Grain screened', fontsize=14)

sns.violinplot(x='Wax', y='Grain screened', data=df_copy, ax=ax)
ax.set_xlabel('Wax', size=12, alpha=0.8)
ax.set_ylabel('Grain screened', size=12, alpha=0.8)
```

Out[94]:
```
Text(0, 0.5, 'Grain screened')
```



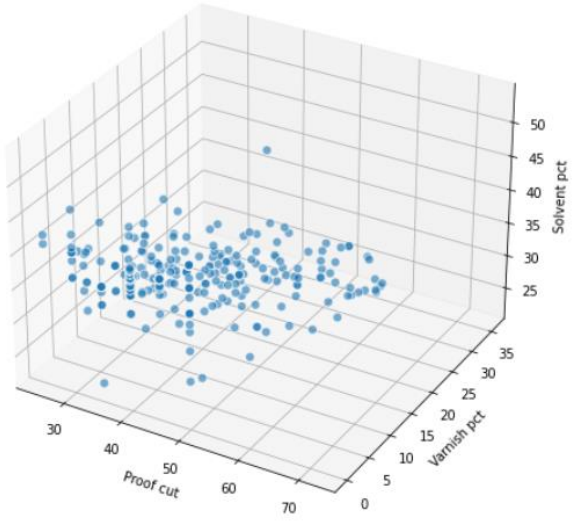
Visualizing data in One Dimension (3D)

In [95]:
```
# Leveraging the concepts of hue for categorical dimension
ax = sns.kdeplot(df_copy['Solvent pct'], df_copy['Ink pct'],
                 cmap="YlOrBr", shade=True, shade_lowest=False)
```



In [96]:
```
# Visualizing 3-D numeric data with Scatter Plots

fig = plt.figure(figsize=(12, 8))
ax = fig.add_subplot(111, projection='3d')

xs = df_copy['Proof cut']
ys = df_copy['Varnish pct']
zs = df_copy['Solvent pct']

ax.scatter(xs, ys, zs, s=50, alpha=0.6, edgecolors='w')


ax.set_xlabel('Proof cut')
ax.set_ylabel('Varnish pct')
ax.set_zlabel('Solvent pct')
```

Out[96]:
```
Text(0.5, 0, 'Solvent pct')
```


Solvent pct
50
45
40
35
30
25
Varnish pct
35
30
25
20
15
10
5
0
Proof cut
30
40
50
60
70


jupyter