Министерство образования Республики Беларусь

УО «Брестский государственный технический университет»

Кафедра ИИТ

**Лабораторная работа №8**

По дисциплине: “Языки программирования”

Тема: «Python. Основы Pandas»

**Вариант №6**

**Выполнил**: студент 2 курса

группы ПО-7

Рекун И.В.

**Проверила:** Бойко Д.О.

Брест 2021

### Цель работы

Ознакомиться с основами библиотеки pandas и научиться строить графики с использованием библиотек matplotlib.pyplot и seaborn.

### Общие требования

1. Загрузить датасет в pandas и проверить на доступность

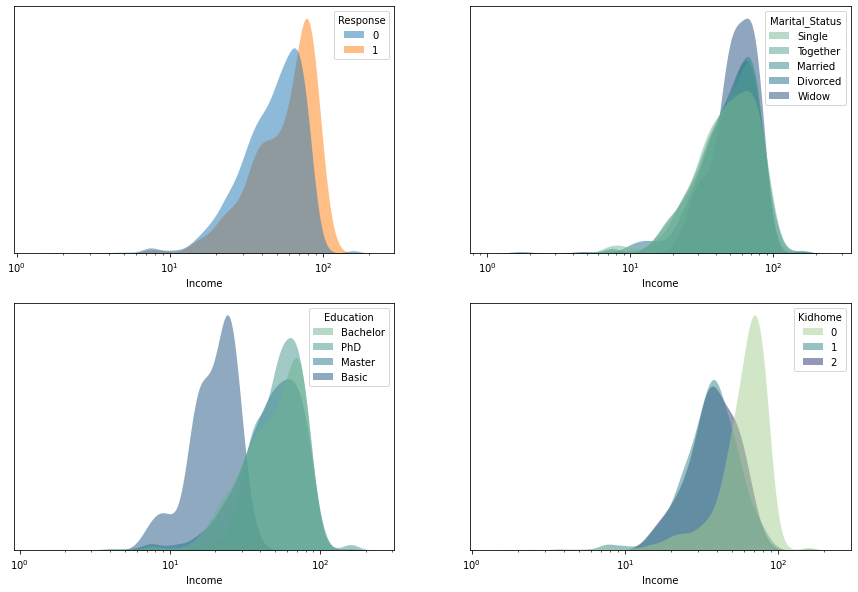
2. Вывести общую информацию о датасете

3. Проверка наличия NULL-данных. При их наличии вывести на экран

4. Удалить колонки "Z\_CostContact", "Z\_Revenue"

5. Переименовать колонку "Year\_Birth" в "Age"

6. Оценить состояние колонок "Marital\_Status", "Education". Построить информативные диаграммы и гистограммы для каждой.

1. Создать гистаграмму по колонке "Age" и оценить на распределение по Гауссу.
2. Оценка полей "Kidhome" и "Teenhome", "Response" и "Income" (диаграммы и гистограммы)
3. Построить графики "Response", "Marital\_Status", "Education" и  "Kidhome" пообразцу
4. Построить heatmap для всех числовых колонок:

### Решение:

**№1**

df = pd.read\_csv("dataset.csv", delimiter="\t")

**№2**

df.info()

**№3**

df[df.isnull().T.any()]

**№4**

df.drop(["Z\_CostContact", "Z\_Revenue"], axis=1, inplace=True)

**№5**

df.rename({"Year\_Birth": "Age"}, axis=1, inplace=True)

**№6**

ms\_df\_c = df["Marital\_Status"].value\_counts().to\_frame()

ms\_df = ms\_df\_c[ms\_df\_c["Marital\_Status"] > 50]

ed\_df = df["Education"].value\_counts().to\_frame()

ms\_df[ms\_df["Marital\_Status"] > 50].plot(

kind='pie',

autopct='%1.1f%%',

legend=None,

ylabel="",

subplots=True,

shadow=True,

explode=(0.13, 0.1, 0.1, 0.25, 0.15),

title="Marital Status",

figsize = (6,6)

)

plt.show()

ed\_df.plot(

y="Education",

kind='pie',

autopct='%1.1f%%',

legend=None,

ylabel="",

shadow=True,

explode=(0.13, 0.1, 0.1, 0.25, 0.15),

title="Education",

figsize = (6,6)

)

plt.show()

sns.histplot(data=df, x="Marital\_Status", stat='percent', hue="Response")

plt.show()

sns.histplot(data=df, x="Education", stat='percent', hue="Response")

plt.show()

**№7**

def calculate\_age(born) -> int:

return int(datetime.today().strftime("%Y")) - int(born)

df["Age"] = df["Age"].apply(calculate\_age)

filtered\_df = df[df["Age"] < 100]

sns.histplot(x=filtered\_df["Age"], kde=True)

plt.show()

**№8**

def show\_bar(data\_name: str) -> None:

df[data\_name].value\_counts(normalize=True).mul(100).to\_frame().plot(

kind='bar',

legend=None,

xlabel=data\_name,

ylabel="Percent"

)

plt.show()

show\_bar("Kidhome")

show\_bar("Teenhome")

show\_bar("Response")

sns.kdeplot(df["Income"], shade=True, clip=(10\*\*0, 10\*\*5.1))

plt.show()

**№9**

def show\_kde(hue: str, clip: tuple = (10\*\*0, 10\*\*5.05)):

plt.figure(figsize = (7, 7))

sns.kdeplot(data=df, x="Income", hue=hue, shade=True, clip=clip)

plt.show()

show\_kde("Response")

show\_kde("Marital\_Status")

show\_kde("Education")

show\_kde("Kidhome")

**№10**

numerics = ['int16', 'int32', 'int64', 'float16', 'float32', 'float64']

newdf = df.select\_dtypes(include=numerics)

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

sns.heatmap(newdf.corr(), cmap='coolwarm', annot=True, fmt='.1g')

plt.show()