Федеральное государственное бюджетное образовательное учреждение высшего образования «Московский государственный технический университет имени Н.Э. Баумана (национальный исследовательский университет)» (МГТУ им. Н.Э. Баумана)

Факультет «Информатика и вычислительная техника»
Кафедра ИУ5 «Системы обработки информации и управления»

Курс «Технология машинного обучения»

Отчет по лабораторной работе №2
«Обработка пропусков в данных, кодирование категориальных признаков и масштабирование данных»

Выполнил:
студент группы ИУ5-63Б

Баркалова И.В.

Подпись и дата:

Проверил:
преподаватель каф. ИУ5
Гапанюк Ю.Е.

Подпись и дата:

Москва, 2022 г.

Цель лабораторной работы:

Изучение способов предварительной обработки данных для дальнейшего формирования моделей

Описание задания:

- Выбрать набор данных (датасет)
- Для выбранного датасета выполнить следующие задачи:
  1. Обработку пропусков в данных
  2. Кодирование категориальных признаков
  3. Масштабирование данных

Выполнение лабораторной работы:

Для выполнения данной лабораторной работы будем использовать данные банка, который автоматизирует выдачу кредитов своим клиентам.
Для начала импортируем DataSet и посмотрим на "шапку". Параметр sep зададим, чтобы использовать разделитель, отличный от стандартного - в нашем случае это точка с запятой.

In [56]:
```
import pandas as pd
import numpy as np
```

Отметим, что при таком импорте DataSet необходимо разместить файл с DataSet в том же месте (в той же директории и папке), что и проект Jupyter notebook.

In [57]:
```
df = pd.read_csv('bank-full1.csv', sep=';')
```

In [58]:
```
df.head()
```

Out[58]:

	Возраст	Работа	Семейный статус	Образование	Кредитный дефолт	Ипотека	Займ	Контакт	Месяц	День недели	...	Кампания	День	Предыдущ... конта
0	27	Самозанятый	Не женат / не замужем	Университетская степень	Нет	Нет	Нет	Городской телефон	Октябрь	Пятница	...	1	999	
1	30	Предприниматель	Женат / замужем	Университетская степень	Нет	Нет	Нет	Городской телефон	Октябрь	Пятница	...	1	999	
2	39	Голубой воротничок	Женат / замужем	Базовое (9 классов)	Нет	Нет	Нет	Городской телефон	Октябрь	Пятница	...	1	999	
3	42	Менеджер	Женат / замужем	Высшая школа	Нет	Да	Да	Городской телефон	Октябрь	Пятница	...	1	999	
4	42	Самозанятый	Женат / замужем	Базовое (4 класса)	Нет	Нет	Нет	Городской телефон	Октябрь	Пятница	...	1	999	

5 rows × 21 columns

Для начала удалим дубликаты, так как дублирующие записи искажают показатели DataSet. Но перед удалением дубликатов обязательно узнаем, сколько записей находится в DataSet.

In [59]:
```
df.shape
```

Out[59]: (41188, 21)

Видно, что Pandas нашел и удалил 12 дубликатов.

In [60]:
```
df = df.drop_duplicates()
df.shape
```

Out[60]: (41176, 21)

Обработка пропусков!
Помним,что если у признака более 70% пропусков, то такой признак удаляют. Поэтому проверим, насколько наши признаки полны.

In [63]:
```
column_values = df[['Работа', 'Семейный статус', 'Образование', 'Контакт', 'Месяц', 'День недели', 'Доходность']].values.rave
unique_values =  pd.unique(column_values)
print(unique_values)
```

```
['Самозанятый' 'Не женат / не замужем' 'Университетская степень'
 'Городской телефон' 'Октябрь' 'Пятница' 'Отсутствует' 'Предприниматель'
 'Женат / замужем' 'Голубой воротничок' 'Базовое (9 классов)' 'Менеджер'
 'Высшая школа' 'Базовое (4 класса)' 'Техник' 'Профессиональный курс'
 'Разведен(-а)' 'Неизвестно' 'Сотовый телефон' 'Август' 'Понедельник'
 'Студент' 'Домохозяйка' 'Обслуживающий персонал' 'Базовое (6 классов)'
 'Пенсионер' 'Четверг' 'Вторник' 'Не присутствует' 'Июль' 'Среда' 'Июнь'
 'Неграмотный' 'Май' 'Ноябрь' 'Присутствует' 'Самозанятый' 'Декабрь'
 'Март' 'Апрель' 'Сентябрь']
```

Существует несколько способов обозначить пропуски, и зачастую создатели датасета не описывают данные в достаточной мере, и определять, как обозначены пропуски, приходится вручную.
Например:

1) NaN / NaT (упрощенно: "не число" / "не время")
2)Пустая ячейка
3) Для числовых признаков – радикальный выброс. К примеру, для столбца "День" это число 999.
4) Маркер или нестандартный символ

Встроенные методы Pandas позволяют с легкостью справиться с первыми двумя разновидностями таких пробелов. Разберемся для начала с категориальными переменными, объединив их в один список.

In [65]:
```
df = df.replace({"День": 999,
                 "Работа": "Неизвестно",
                 "Семейный статус": "Неизвестно",
                 "Образование": "Неизвестно",
                 "Месяц": "Неизвестно",
                 "День недели": "Неизвестно",
                 "Доходность": "Неизвестно",
                 }, np.nan)

df.head()
```

Out[65]:

	Возраст	Работа	Семейный статус	Образование	Кредитный дефолт	Ипотека	Займ	Контакт	Месяц	День недели	...	Кампания	День	Предыдущ конта
0	27	Самозанятый	Не женат / не замужем	Университетская степень	Нет	Нет	Нет	Городской телефон	Октябрь	Пятница	...	1	NaN	
1	30	Предприниматель	Женат / замужем	Университетская степень	Нет	Нет	Нет	Городской телефон	Октябрь	Пятница	...	1	NaN	
2	39	Голубой воротничок	Женат / замужем	Базовое (9 классов)	Нет	Нет	Нет	Городской телефон	Октябрь	Пятница	...	1	NaN	
3	42	Менеджер	Женат / замужем	Высшая школа	Нет	Да	Да	Городской телефон	Октябрь	Пятница	...	1	NaN	
4	42	Самозанятый	Женат / замужем	Базовое (4 класса)	Нет	Нет	Нет	Городской телефон	Октябрь	Пятница	...	1	NaN	

5 rows × 21 columns

Среди всех признаков 96% пропусков находится у признака "День", поэтому он поделжит удалению.

In [67]:
```
df.isnull().mean() * 100
```

Out[67]:
```
Возраст                              0.000000
Работа                               0.801438
Семейный статус                      0.194288
Образование                          4.201477
Кредитный дефолт                     0.000000
Ипотека                              0.000000
Займ                                 0.000000
Контакт                              0.000000
Месяц                                0.000000
День недели                          0.000000
Длительность                         0.000000
Кампания                             0.000000
День                                96.320672
Предыдущий контакт                   0.000000
Доходность                           0.000000
Колебание уровня безработицы         0.000000
Индекс потребительских цен           0.000000
Индекс потребительской уверенности   0.000000
Европейская межбанковская ставка     0.000000
Количество сотрудников в компании    0.000000
y                                    0.000000
dtype: float64
```

In [68]:
```
df = df.drop(columns=['День'])
df.head()
```

Out[68]:

	Возраст	Работа	Семейный статус	Образование	Кредитный дефолт	Ипотека	Займ	Контакт	Месяц	День недели	Длительность	Кампания	Предь
0	27	Самозанятый	Не женат / не замужем	Университетская степень	Нет	Нет	Нет	Городской телефон	Октябрь	Пятница	24	1	
1	30	Предприниматель	Женат / замужем	Университетская степень	Нет	Нет	Нет	Городской телефон	Октябрь	Пятница	223	1	
2	39	Голубой воротничок	Женат / замужем	Базовое (9 классов)	Нет	Нет	Нет	Городской телефон	Октябрь	Пятница	270	1	
3	42	Менеджер	Женат / замужем	Высшая школа	Нет	Да	Да	Городской телефон	Октябрь	Пятница	103	1	
4	42	Самозанятый	Женат / замужем	Базовое (4 класса)	Нет	Нет	Нет	Городской телефон	Октябрь	Пятница	478	1	

Процесс обработки пропусков, к счастью, можно сократить с помощью sklearn.impute.SimpleImputer. Мы выбираем все категориальные переменные и применяем стратегию "[вставить вместо пропуска] самое распространенное значение":

In [69]:
```
from sklearn.impute import SimpleImputer
imputer = SimpleImputer(missing_values = np.nan, strategy = 'most_frequent')
df["Работа"] = imputer.fit_transform(df["Работа"].values.reshape(-1,1))[:,0]
df["Семейный статус"] = imputer.fit_transform(df["Семейный статус"].values.reshape(-1,1))[:,0]
df["Образование"] = imputer.fit_transform(df["Образование"].values.reshape(-1,1))[:,0]
df["Месяц"] = imputer.fit_transform(df["Месяц"].values.reshape(-1,1))[:,0]
df["День недели"] = imputer.fit_transform(df["День недели"].values.reshape(-1,1))[:,0]
df["Доходность"] = imputer.fit_transform(df["Доходность"].values.reshape(-1,1))[:,0]
```

Признаки, принадлежащие к булевому типу данных, обрабатываются алгоритмом тем же образом. Целевую переменную Y мы не обрабатываем (если в этом столбце есть пропуски, такие строки стоит удалить):

In [70]:
```
imputer = SimpleImputer(missing_values = np.nan, strategy = 'most_frequent')
df["Кредитный дефолт"] = imputer.fit_transform(df["Кредитный дефолт"].values.reshape(-1,1))[:,0]
df["Ипотека"] = imputer.fit_transform(df["Ипотека"].values.reshape(-1,1))[:,0]
df["Займ"] = imputer.fit_transform(df["Займ"].values.reshape(-1,1))[:,0]
```

Подобным образом заполняются пустоты в числовых переменных, только стратегия теперь – "вставить среднее значение".

In [71]:
```
imputer = SimpleImputer(missing_values = np.nan, strategy = 'mean')
df["Возраст"] = imputer.fit_transform(df["Возраст"].values.reshape(-1,1))[:,0]
df["Длительность"] = imputer.fit_transform(df["Длительность"].values.reshape(-1,1))[:,0]
df["Кампания"] = imputer.fit_transform(df["Кампания"].values.reshape(-1,1))[:,0]
df["Предыдущий контакт"] = imputer.fit_transform(df["Предыдущий контакт"].values.reshape(-1,1))[:,0]
df["Колебание уровня безработицы"] = imputer.fit_transform(df["Колебание уровня безработицы"].values.reshape(-1,1))[:,0]
df["Индекс потребительских цен"] = imputer.fit_transform(df["Индекс потребительских цен"].values.reshape(-1,1))[:,0]
df["Индекс потребительской уверенности"] = imputer.fit_transform(df["Индекс потребительской уверенности"].values.reshape(-1,1
df["Европейская межбанковская ставка"] = imputer.fit_transform(df["Европейская межбанковская ставка"].values.reshape(-1,1))[:
df["Количество сотрудников в компании"] = imputer.fit_transform(df["Количество сотрудников в компании"].values.reshape(-1,1)
```

In [72]:
```
df.head()
```

Out[72]:

	Возраст	Работа	Семейный статус	Образование	Кредитный дефолт	Ипотека	Займ	Контакт	Месяц	День недели	Длительность	Кампания	Предь
0	27.0	Самозанятый	Не женат / не замужем	Университетская степень	Нет	Нет	Нет	Городской телефон	Октябрь	Пятница	24.0	1.0	
1	30.0	Предприниматель	Женат / замужем	Университетская степень	Нет	Нет	Нет	Городской телефон	Октябрь	Пятница	223.0	1.0	
2	39.0	Голубой воротничок	Женат / замужем	Базовое (9 классов)	Нет	Нет	Нет	Городской телефон	Октябрь	Пятница	270.0	1.0	
3	42.0	Менеджер	Женат / замужем	Высшая школа	Нет	Да	Да	Городской телефон	Октябрь	Пятница	103.0	1.0	
4	42.0	Самозанятый	Женат / замужем	Базовое (4 класса)	Нет	Нет	Нет	Городской телефон	Октябрь	Пятница	478.0	1.0	

In [ ]:
```
```