

**Московский государственный технический
университет им. Н.Э. Баумана.**

Факультет «Информатика и управление»

Кафедра «Системы обработки информации и управления»

Отчет по лабораторной работе №2

«Обработка пропусков в данных, кодирование категориальных признаков,
масштабирование данных.»

«Курса «Технологии машинного обучения»»

Выполнила:
студентка группы ИУ5-64
Светашева Ю.В
Подпись и дата:



Проверил:
преподаватель каф. ИУ5
Антонов С.К.
Подпись и дата:

Москва, 2022 г.












Описание задания

1. Выбрать набор данных (датасет), содержащий категориальные признаки и пропуски в данных. Для выполнения следующих пунктов можно использовать несколько различных наборов данных (один для обработки пропусков, другой для категориальных признаков и т.д.)
2. Для выбранного датасета (датасетов) на основе материалов лекции решить следующие задачи:
 - обработку пропусков в данных;
 - кодирование категориальных признаков;
 - масштабирование данных.

Текст программы и экранные формы

 jupyter Lab2_Svetasheva_IU5-64B Last Checkpoint: час назад (autosaved)  Logout

File Edit View Insert Cell Kernel Widgets Help Not Trusted Python 3 (ipykernel) C

          Code 

Для выполнения данной лабораторной работы будем использовать данные пассажиров Титаника

```
In [4]: 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 [5]: df = pd.read_csv('titanic.csv')
```

```
In [5]: df.head()
```

```
Out[5]:
```

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	NaN	S
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	C85	C
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250	NaN	S
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000	C123	S
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500	NaN	S

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

```
In [6]: df.shape
```

```
Out[6]: (891, 12)
```

Видно, что Pandas не нашел дубликатов.

```
In [7]: df = df.drop_duplicates()
df.shape
```

```
Out[7]: (891, 12)
```

Обработка пропусков!

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

```
In [8]: column_values = df[['PassengerId', 'Survived', 'Pclass', 'Name', 'Sex', 'Age', 'SibSp', 'Parch', 'Ticket', 'Fare', 'Cabin', 'Embarked']].values
unique_values = pd.unique(column_values)
print(unique_values)
```

```
[1 0 3 ... 891 'Dooley, Mr. Patrick' '370376']
```

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 [9]: df = df.replace({"PassengerId" : 0,
                        "Survived": "Unknown",
                        "Pclass": "Unknown",
                        "Name": "Unknown",
                        "Sex": "Unknow",
                        "Age": "Unknown",
                        "SibSp": "Unknown",
                        "Parch": "Unknow",
                        "Ticket": "Unknown",
                        "Fare": "Unknown",
                        "Cabin": "Unknow",
                        "Embarked": "Unknow"
                        }, np.nan)

df.head()
```

Out[9]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	NaN	S
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	C85	C
2	3	1	3	Heikinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250	NaN	S
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000	C123	S
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500	NaN	S

Среди всех признаков 96% пропусков находится у признака "Cabin", поэтому он подлежит удалению.

```
In [10]: df.isnull().mean() * 100
```

```
Out[10]: PassengerId    0.000000
Survived    0.000000
Pclass      0.000000
Name        0.000000
Sex         0.000000
Age        19.865320
SibSp       0.000000
Parch       0.000000
Ticket      0.000000
Fare        0.000000
Cabin       77.104377
Embarked    0.224467
dtype: float64
```

```
In [11]: df = df.drop(columns=['Cabin'])
df.head()
```

Out[11]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Embarked
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	S
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	C
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250	S
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000	S
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500	S

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". Обращаем только те колонки, в которых выявили наличие пропусков на предыдущем шаге:

```
In [12]: from sklearn.impute import SimpleImputer

imputer = SimpleImputer(missing_values = np.nan, strategy = 'most_frequent')

df["Age"] = imputer.fit_transform(df["Age"].values.reshape(-1,1))[:,0]

df["Embarked"] = imputer.fit_transform(df["Embarked"].values.reshape(-1,1))[:,0]
```

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

```
In [13]: imputer = SimpleImputer(missing_values = np.nan, strategy = 'mean')

df["Age"] = imputer.fit_transform(df["Age"].values.reshape(-1,1))[:,0]
```

In [14]: df.head()

Out[14]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Embarked
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	S
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	C
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250	S
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000	S
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500	S