Московский государственный технический университет им. Н.Э. Баумана

Факультет «Информатика и системы управления»

Кафедра ИУ5 «Системы обработки информации и управления

Рубежный контроль 1

Вариант №10

Выполнила:

студентка группы ИУ5-22м

Миронова Александра

Задание

Каждая задача предполагает использование набора данных. Набор данных выбирается Вами произвольно с учетом следующих условий:

- Вы можете использовать один набор данных для решения всех задач, или решать каждую задачу на своем наборе данных.
- Набор данных должен отличаться от набора данных, который использовался в лекции для решения рассматриваемой задачи.
- Вы можете выбрать произвольный набор данных (например тот, который Вы использовали в лабораторных работах) или создать собственный набор данных (что актуально для некоторых задач, например, для задач удаления псевдоконстантных или повторяющихся признаков).
- Выбранный или созданный Вами набор данных должен удовлетворять условиям поставленной задачи. Например, если решается задача устранения пропусков, то набор данных должен содержать пропуски.

Задача 1

Для набора данных проведите устранение пропусков для одного (произвольного) категориального признака с использованием метода заполнения наиболее распространенным значением.

Задача 2

Для набора данных проведите удаление повторяющихся признаков.

Дополнительно

Для произвольной колонки данных построить гистограмму.

Импортируем библиотеки

```
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np
from sklearn.impute import MissingIndicator, SimpleImputer
import seaborn as sns


def plot_hist_diff(old_ds, new_ds, cols):
    """
    Разница между распределениями до и после устранения пропусков
    """
    for c in cols:
        fig = plt.figure()
        ax = fig.add_subplot(111)
        ax.title.set_text('Поле - ' + str(c))
        old_ds[c].hist(bins=50, ax=ax, density=True, color='green')
        new_ds[c].hist(bins=50, ax=ax, color='blue', density=True, alpha=0.5)
        plt.show()
```

Загружаем набор данных

```
df_task_1 = pd.read_csv('airport.csv')
df_task_1
```

	id str	ident str	type str	name str	latitude_deg float	longitude_deg float	elevation
0	6523	00A	heliport	Total Rf Heliport	40.070801	-74.933601	'
1	323361	00AA	small_airport	Aero B Ranch Airport	38.704022	-101.473911	343
2	6524	00AK	small_airport	Lowell Field	59.949200	-151.695999	45
3	6525	00AL	small_airport	Epps Airpark	34.864799	-86.770302	82
4	6526	00AR	closed	Newport Hospital & Clinic Heliport	35.608700	-91.254898	23
...	...	...	...	...	...	...	
55095	317861	ZYYK	medium_airport	Yingkou Lanqi Airport	40.542524	122.358600	
55096	32753	ZYYY	medium_airport	Shenyang Dongta Airport	41.784401	123.496002	N
55097	46378	ZZ-0001	heliport	Sealand Helipad	51.894444	1.482500	4
55098	307326	ZZ-0002	small_airport	Glorioso Islands Airstrip	-11.584278	47.296389	'
55099	313629	ZZZZ	small_airport	Satsuma I?jima Airport	30.784722	130.270556	33

55100 rows × 18 columns

В качестве исходных данных был выбран набор данных аэропорта. Набор данных содержит пропуски в некоторых категориальных колонках. Убедимся в этом.

```
# Колонки с пропусками
df_task_1_na = [c for c in df_task_1.columns if df_task_1[c].isnull().sum() > 0]
df_task_1_na
```

```
['elevation_ft int',
 'continent str',
 'iso_country str',
 'municipality str',
 'gps_code str',
 'iata_code str',
 'local_code str',
 'home_link str',
 'wikipedia_link str',
 'keywords str']
```

```
# Доля (процент) пропусков
[(c, df_task_1[c].isnull().mean()) for c in df_task_1_na]
```

```
[('elevation_ft int', 0.12548094373865698),
 ('continent str', 0.5038112522686026),
 ('iso_country str', 0.004482758620689655),
 ('municipality str', 0.10266787658802178),
 ('gps_code str', 0.255989110707804),
 ('iata_code str', 0.8333756805807623),
 ('local_code str', 0.47778584392014517),
 ('home_link str', 0.9454990925589837),
 ('wikipedia_link str', 0.8212704174228675),
 ('keywords str', 0.8294918330308529)]
```

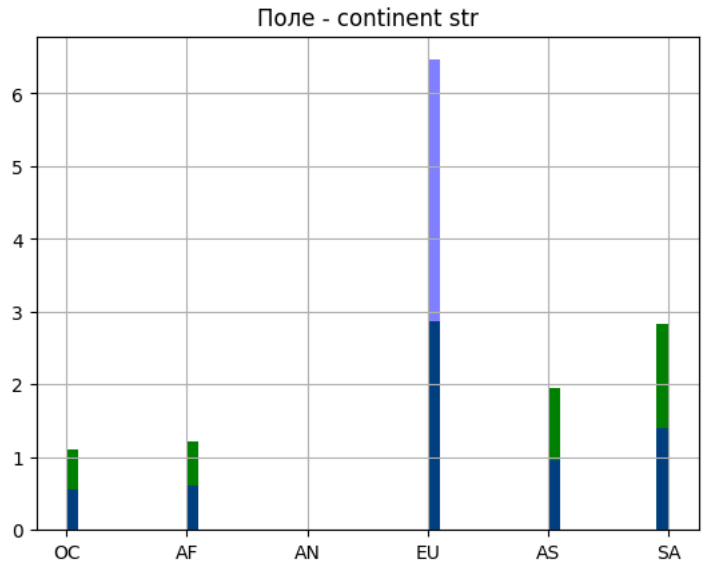
Задание 1

Для устранения пропусков наиболее распространенным значением выберем категориальный признак аббривеатуры континента "continent str". Данные этого признака не являются главной информацией о полете, если возникнут неточности в результате заполнения пустых значений, более точную информацию о пункте отлета и прилета можно унать в признаке "iso_region str".

```
cat_col=['continent str']
cat_new = df_task_1[cat_col].copy()
```

```
def impute_column(dataset, column, strategy_param, fill_value_param=None):
    """
    Заполнение пропусков в одном признаке
    """
    temp_data = dataset[[column]].values
    size = temp_data.shape[0]

    indicator = MissingIndicator()
    mask_missing_values_only = indicator.fit_transform(temp_data)

    imputer = SimpleImputer(strategy=strategy_param,
                            fill_value=fill_value_param)
    all_data = imputer.fit_transform(temp_data)

    missed_data = temp_data[mask_missing_values_only]
    filled_data = all_data[mask_missing_values_only]

    return all_data.reshape((size,)), filled_data, missed_data
```

```
elevation_ft_cat_new, _, _ = impute_column(cat_new, 'continent str', 'most_frequent')
cat_new['continent str'] = elevation_ft_cat_new
plot_hist_diff(df_task_1, cat_new, cat_col)
```



Задание 2

При визуализации части данных набора в начале первого задания была заметна схожесть значений признаков "gps_code str" и "ident str". Проверим, насколько они отличаются.

```
gps_col= df_task_1['gps_code str'].copy()
ident_col = df_task_1['ident str'].copy()
gps_df = pd.DataFrame(gps_col)
gps_df
```

	gps_code str
0	00A
1	00AA
2	00AK
3	00AL
4	NaN
...	...
55095	ZYYK
55096	ZYYY
55097	NaN
55098	NaN
55099	RJX7

55100 rows × 1 columns

```
mismatch = 0
for i in range(len(df_task_1)):
    if (gps_col[i] != ident_col[i]):
        mismatch=mismatch+1
1-mismatch/len(df_task_1)
```

```
0.6993647912885663
```

Мы видим, что значения столбцов совпадают на 70 процентов. Причем 26% всех записей признака "gps_code str" просто пропущены. Скорее всенго именно поэтому несовпадение по столбцам около 30%. Эти наблюдения позволяют принять решение об удалении столбца "gps_code str".

```
df_task_2 = df_task_1.drop(['gps_code str'], axis=1)
df_task_2
```

	id str	ident str	type str	name str	latitude_deg float	longitude_deg float	elevation_ft int	continent str	iso_country str	iso_region str	municip
0	6523	00A	heliport	Total Rf Heliport	40.070801	-74.933601	11.0	NaN	US	US-PA	Ber
1	323361	00AA	small_airport	Aero B Ranch Airport	38.704022	-101.473911	3435.0	NaN	US	US-KS	
2	6524	00AK	small_airport	Lowell Field	59.949200	-151.695999	450.0	NaN	US	US-AK	Ancho
3	6525	00AL	small_airport	Epps Airpark	34.864799	-86.770302	820.0	NaN	US	US-AL	H
4	6526	00AR	closed	Newport Hospital & Clinic Heliport	35.608700	-91.254898	237.0	NaN	US	US-AR	N
...	...	...	...	...	...	...	...	...	...	...	
55095	317861	ZYYK	medium_airport	Yingkou Lanqi Airport	40.542524	122.358600	0.0	AS	CN	CN-21	Y
55096	32753	ZYYY	medium_airport	Shenyang Dongta Airport	41.784401	123.496002	NaN	AS	CN	CN-21	She
55097	46378	ZZ-0001	heliport	Sealand Helipad	51.894444	1.482500	40.0	EU	GB	GB-ENG	S
55098	307326	ZZ-0002	small_airport	Glorioso Islands Airstrip	-11.584278	47.296389	11.0	AF	TF	TF-U-A	(Gl
55099	313629	ZZZZ	small_airport	Satsuma I?jima Airport	30.784722	130.270556	338.0	AS	JP	JP-46	Mishim

55100 rows × 17 columns

Дополнительное задание

+ Код + Текст

Построим гистограмму для признака "type str".

```
sns.barplot(df_task_1, y="type str", x="id str", estimator="sum", width=0.5, orient="y")
```

```
<Axes: xlabel='id str', ylabel='type str'>
```