

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

Отчёт по рубежному контролю №2 по курсу «Технологии машинного обучения».

«Методы построения моделей машинного обучения».

Выполнил:
Заузолков Д. А.
студент группы ИУ5-62Б

Проверил:
Гапанюк Ю.Е.

Подпись и дата:

Подпись и дата:

Москва, 2022

1. Формулировка задачи:

Задание. Для заданного набора данных (по Вашему варианту) постройте модели классификации или регрессии (в зависимости от конкретной задачи, рассматриваемой в наборе данных). Для построения моделей используйте методы 1 и 2 (по варианту для Вашей группы). Оцените качество моделей на основе подходящих метрик качества (не менее двух метрик). Какие метрики качества Вы использовали и почему? Какие выводы Вы можете сделать о качестве построенных моделей? Для построения моделей необходимо выполнить требуемую предобработку данных: заполнение пропусков, кодирование категориальных признаков, и т.д.

Метод 1 - Линейная/логистическая регрессия Метод 2 - Случайный лес Набор данных:
<https://www.kaggle.com/jessemostipak/hotel-booking-demand>

2. Текстовое описание набора данных

В качестве предметной области будем рассматривать процесс бронирования номера в отеле/гостинице. В датасете содержится подробная информация о бронировании номера гостем:

- hotel - Тип отеля (курортный или городской)
- is_canceled - Булево значение, указывающее, было ли бронирование отменено (1) или нет (0)
- lead_time - Количество дней, прошедших между датой ввода бронирования в систему управления гостиницей и датой прибытия
- arrival_date_year - Год приезда
- arrival_date_month - Месяц прибытия
- arrival_date_week_number - Номер недели в году прибытия
- arrival_date_day_of_month - День прибытия
- stays_in_weekend_nights - Количество ночей в выходные (суббота или воскресенье), которые гость проживал или забронировал
- stays_in_week_nights - Количество ночей в неделю (с понедельника по пятницу), которые гость останавливался или забронировал
- adults - Количество взрослых
- children - Количество детей
- babies - Количество младенцев
- meal - Тип забронированного питания. Категории представлены в стандартных пакетах гостеприимства: Undefined / SC - пакет без питания; BB - кровать и завтрак; HB - полупансион (завтрак и еще один прием пищи - обычно ужин); FB - Полный пансион (завтрак, обед и ужин)
- country - страна в формате ISO 3155-3: 2013.
- market_segment - Обозначение сегмента рынка. В категориях термин «ТА» означает «Туристические агенты», а «ТО» означает «Туроператоры».
- distribution_channel - Канал распространения бронирования. Термин «ТА» означает «Туристические агенты», а «ТО» означает «Туроператоры».
- is_repeated_guest - Значение, указывающее, было ли имя бронирования от повторного гостя (1) или нет (0)
- previous_cancellations - Количество предыдущих бронирований, которые были отменены клиентом до текущего бронирования.
- previous_bookings_not_canceled - Количество предыдущих бронирований, которые не были отменены клиентом до текущего бронирования
- reserved_room_type - Код типа зарезервированного номера

- `assigned_room_type` - Код для типа номера, назначенного для бронирования. Иногда на- значенный тип номера отличается от типа зарезервированного номера по причинам, свя- занным с работой отеля (например, из-за избыточного бронирования) или по запросу кли- ента. Код представлен вместо обозначения из соображений анонимности
- `booking_changes` - Количество изменений / дополнений, внесенных в бронирование с мо- мента внесения бронирования в систему управления гостиницей до момента заселения или отмены
- `deposit_type` - Индикация того, внес ли клиент предоплату для гарантии бронирования. Эта переменная может принимать три категории: `No Deposit` - депозит не был внесен; `Non Refund` - внесен залог в размере полной стоимости проживания; `Refundable` - был внесен депозит в размере, меньшем общей стоимости проживания.
- `agent` - ID туристического агентства, сделавшего бронирование
- `company` - ID компании / юридического лица, совершившего бронирование или ответствен- ного за его оплату.
- `days_in_waiting_list` - Количество дней, в течение которых бронирование находилось в ли- сте ожидания, прежде чем оно было подтверждено клиенту.
- `customer_type` - Тип бронирования, предполагающий одну из четырех категорий: `Contract` - когда с бронированием связано выделение или другой тип контракта; `Group` - когда бро- нирование связано с группой; `Transient` - когда бронирование не является частью группы или контракта и не связано с другим временным бронированием; `Transient-party` - когда бронирование является временным, но связано как минимум с другим временным брони- рованием.
- `adr` - Число представляет собой средний доход от аренды одной оплачиваемой занятой комнаты в заданный период времени (общее количество ночей проживания)
- `required_car_parking_spaces` - Количество мест для машин, требуемых заказчиком
- `total_of_special_requests` - Количество особых запросов, сделанных клиентом (например, две односпальные кровати или высокий этаж)
- `reservation_status` - Статус последнего бронирования, допускающий одну из трех катего- рий: `Canceled` - бронирование было отменено клиентом; `Check-Out` - клиент зарегистри- ровался, но уже уехал; `No-Show` - клиент не заселился и не проинформировал отель о причине
- `reservation_status_date` - Дата, когда был установлен последний статус. Эта переменная может использоваться вместе с `ReservationStatus`, чтобы понять, когда было отменено бронирование или когда клиент выписался из отеля.

3. Основные характеристики датасета

```
[36]: import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
import warnings
warnings.filterwarnings('ignore')
```

```
[37]: data = pd.read_csv('hotel_bookings.csv')
```

Выведем первые 5 строк датасета:

```
[38]: data.head()
```

```
[38]:      hotel  is_canceled  lead_time  arrival_date_year  arrival_date_month \
0  Resort Hotel           0         342             2015              July
1  Resort Hotel           0         737             2015              July
2  Resort Hotel           0           7             2015              July
```

3	Resort Hotel	0	13	2015	July
4	Resort Hotel	0	14	2015	July

	arrival_date_week_number	arrival_date_day_of_month \
0	27	1
1	27	1
2	27	1
3	27	1
4	27	1

	stays_in_weekend_nights	stays_in_week_nights	adults ...	deposit_type \
0	0		0	2 ... No Deposit
1	0		0	2 ... No Deposit
2	0		1	1 ... No Deposit
3	0		1	1 ... No Deposit
4	0		2	2 ... No Deposit

	agent	company	days_in_waiting_list	customer_type	adr \
0	NaN	NaN	0	Transient	0.0
1	NaN	NaN	0	Transient	0.0
2	NaN	NaN	0	Transient	75.0
3	304.0	NaN	0	Transient	75.0
4	240.0	NaN	0	Transient	98.0

	required_car_parking_spaces	total_of_special_requests	reservation_status \
0	0		0 Check-Out
1	0		0 Check-Out
2	0		0 Check-Out
3	0		0 Check-Out
4	0		1 Check-Out

	reservation_status_date 0
	2015-07-01
1	2015-07-01
2	2015-07-02
3	2015-07-02
4	2015-07-03

[5 rows x 32 columns]

Определим размер датасета:

[39]: data.shape

[39]: (119390, 32)

Определим типы столбцов:

[40]: data.dtypes

[40] :	hotel	object
	is_canceled	int64
	lead_time	int64
	arrival_date_year	int64
	arrival_date_month	object

arrival_date_week_number	int64
arrival_date_day_of_month	int64
stays_in_weekend_nights	int64
stays_in_week_nights	int64
adults	int64
children	float64
babies	int64
meal	object
country	object
market_segment	object
distribution_channel	object
is_repeated_guest	int64
previous_cancellations	int64
previous_bookings_not_canceled	int64
reserved_room_type	object
assigned_room_type	object
booking_changes	int64
deposit_type	object
agent	float64
company	float64
days_in_waiting_list	int64
customer_type	object
adr	float64
required_car_parking_spaces	int64
total_of_special_requests	int64
reservation_status	object
reservation_status_date	object
dtype:	object

[41]: Выведем список параметров датасета и для каждого найдём количество null значений:

```
for column in data.columns:
    print(f'{column}: {data[column].isnull().sum()} null values')
```

```
hotel: 0 null values is_canceled: 0 null
values lead_time: 0 null values
arrival_date_year: 0 null values
arrival_date_month: 0 null values
arrival_date_week_number: 0 null values
arrival_date_day_of_month: 0 null values
stays_in_weekend_nights: 0 null values
stays_in_week_nights: 0 null values adults: 0 null
values
children: 4 null values babies: 0
null values meal: 0 null values
country: 488 null values
market_segment: 0 null values distribution_channel: 0 null
values is_repeated_guest: 0 null values
previous_cancellations: 0 null values
previous_bookings_not_canceled: 0 null values
reserved_room_type: 0 null values
```

assigned_room_type: 0 null values
 booking_changes: 0 null values
 deposit_type: 0 null values agent:
 16340 null values
 company: 112593 null values days_in_waiting_list:
 0 null values customer_type: 0 null values
 adr: 0 null values required_car_parking_spaces: 0 null
 values total_of_special_requests: 0 null values
 reservation_status: 0 null values
 reservation_status_date: 0 null values

Следующие столбцы имеют пустые ячейки: children, country, agent, company. Уберем записи с пустым параметром country и children из датасета, а столбцы agent и company удалим, поскольку они являются неинформативными :

```
[42]: data.drop(data[data['country'].isnull()].index, inplace=True)
      data.drop(data[data['children'].isnull()].index, inplace=True)
      for column in data.columns:
          print(f'{column}: {data[column].isnull().sum()} null values')
```

hotel: 0 null values is_canceled: 0 null
 values lead_time: 0 null values
 arrival_date_year: 0 null values
 arrival_date_month: 0 null values
 arrival_date_week_number: 0 null values
 arrival_date_day_of_month: 0 null values
 stays_in_weekend_nights: 0 null values
 stays_in_week_nights: 0 null values adults: 0 null
 values
 children: 0 null values babies: 0
 null values meal: 0 null values
 country: 0 null values
 market_segment: 0 null values distribution_channel: 0 null
 values is_repeated_guest: 0 null values
 previous_cancellations: 0 null values
 previous_bookings_not_canceled: 0 null values
 reserved_room_type: 0 null values assigned_room_type: 0
 null values booking_changes: 0 null values
 deposit_type: 0 null values agent:
 16004 null values company: 112275
 null values
 days_in_waiting_list: 0 null values customer_type: 0
 null values
 adr: 0 null values required_car_parking_spaces: 0
 null values total_of_special_requests: 0 null values
 reservation_status: 0 null values
 reservation_status_date: 0 null values

```
[43]: data.drop('agent', inplace=True, axis=1)
data.drop('company', inplace=True, axis=1)
data.drop('market_segment', inplace=True, axis=1)
data.drop('distribution_channel', inplace=True, axis=1)
data.drop('reservation_status_date', inplace=True, axis=1)
data.drop('country', inplace=True, axis=1)
data.head()
```

```
[43]:      hotel  is_canceled  lead_time  arrival_date_year  arrival_date_month \
0  Resort Hotel         0        342          2015          July
1  Resort Hotel         0        737          2015          July
2  Resort Hotel         0         7          2015          July
3  Resort Hotel         0        13          2015          July
4  Resort Hotel         0        14          2015          July

      arrival_date_week_number  arrival_date_day_of_month \
0                             27                         1
1                             27                         1
2                             27                         1
3                             27                         1
4                             27                         1

      stays_in_weekend_nights  stays_in_week_nights  adults  ... \
0                             0                     0      2  ...
1                             0                     0      2  ...
2                             0                     1      1  ...
3                             0                     1      1  ...
4                             0                     2      2  ...

      reserved_room_type  assigned_room_type  booking_changes  deposit_type \
0                      C                   C                3  No Deposit
1                      C                   C                4  No Deposit
2                      A                   C                0  No Deposit
3                      A                   A                0  No Deposit
4                      A                   A                0  No Deposit

      days_in_waiting_list  customer_type  adr  required_car_parking_spaces \
0                          0      Transient    0.0                      0
1                          0      Transient    0.0                      0
2                          0      Transient   75.0                      0
3                          0      Transient   75.0                      0
4                          0      Transient   98.0                      0

      total_of_special_requests  reservation_status
0                             0      Check-Out
1                             0      Check-Out
2                             0      Check-Out
3                             0      Check-Out
4                             1      Check-Out
```

[5 rows x 26 columns]

```
[44]: filter = (data.children == 0) & (data.adults == 0) & (data.babies == 0)
data[filter].drop
data = data[~filter]
data
```

```
[44]:
```

	hotel	is_canceled	lead_time	arrival_date_year	\
0	Resort Hotel	0	342	2015	
1	Resort Hotel	0	737	2015	
2	Resort Hotel	0	7	2015	
3	Resort Hotel	0	13	2015	
4	Resort Hotel	0	14	2015	
...	
119385	City Hotel	0	23	2017	
119386	City Hotel	0	102	2017	
119387	City Hotel	0	34	2017	
119388	City Hotel	0	109	2017	
119389	City Hotel	0	205	2017	

	arrival_date_month	arrival_date_week_number	\
0	July	27	
1	July	27	
2	July	27	
3	July	27	
4	July	27	
...	
119385	August	35	
119386	August	35	
119387	August	35	
119388	August	35	
119389	August	35	

	arrival_date_day_of_month	stays_in_weekend_nights	\
0	1	0	
1	1	0	
2	1	0	
3	1	0	
4	1	0	
...	
119385	30	2	
119386	31	2	
119387	31	2	
119388	31	2	
119389	29	2	

	stays_in_week_nights	adults	...	reserved_room_type	\
0	0	2	...	C	
1	0	2	...	C	
2	1	1	...	A	
3	1	1	...	A	
4	2	2	...	A	
...	
119385	5	2	...	A	
119386	5	3	...	E	

119387	5	2 ...	D
119388	5	2 ...	A
119389	7	2 ...	A

	assigned_room_type	booking_changes	deposit_type	\
0	C	3	No Deposit	
1	C	4	No Deposit	
2	C	0	No Deposit	
3	A	0	No Deposit	
4	A	0	No Deposit	
...	
119385	A	0	No Deposit	
119386	E	0	No Deposit	
119387	D	0	No Deposit	
119388	A	0	No Deposit	
119389	A	0	No Deposit	

	days_in_waiting_list	customer_type	adr	\
0	0	Transient	0.00	
1	0	Transient	0.00	
2	0	Transient	75.00	
3	0	Transient	75.00	
4	0	Transient	98.00	
...	
119385	0	Transient	96.14	
119386	0	Transient	225.43	
119387	0	Transient	157.71	
119388	0	Transient	104.40	
119389	0	Transient	151.20	

	required_car_parking_spaces	total_of_special_requests	\
0	0	0	
1	0	0	
2	0	0	
3	0	0	
4	0	1	
...	
119385	0	0	
119386	0	2	
119387	0	4	
119388	0	0	
119389	0	2	

	reservation_status
0	Check-Out
1	Check-Out
2	Check-Out
3	Check-Out
4	Check-Out
...	...
119385	Check-Out
119386	Check-Out
119387	Check-Out

```
119388      Check-Out
119389      Check-Out

[118728 rows x 26 columns]
```

4. Кодирование категориальных признаков

```
[45]: data['hotel'].unique()
```

```
[45]: array(['Resort Hotel', 'City Hotel'], dtype=object)
```

```
[46]: data['arrival_date_month'].unique()
```

```
[46]: array(['July', 'August', 'September', 'October', 'November', 'December', 'January', 'February',
          'March', 'April', 'May', 'June'], dtype=object)
```

```
[47]: data['deposit_type'].unique()
```

```
[47]: array(['No Deposit', 'Refundable', 'Non Refund'], dtype=object)
```

```
[47]: data['customer_type'].unique()
```

```
[51]: array(['Transient', 'Contract', 'Transient-Party', 'Group'], dtype=object)
```

```
[51]: data['reserved_room_type'].unique()
```

```
[52]: array(['C', 'A', 'D', 'E', 'G', 'F', 'H', 'L', 'B'], dtype=object)
```

```
[52]: data['assigned_room_type'].unique()
```

```
[53]: array(['C', 'A', 'D', 'E', 'G', 'F', 'I', 'B', 'H', 'L', 'K'],
          dtype=object)
```

```
[54]: data['meal'].unique()
```

```
[54]: array(['BB', 'FB', 'HB', 'SC', 'Undefined'], dtype=object)
```

```
[56]: data['reservation_status'].unique()
```

```
[56]: array(['Check-Out', 'Canceled', 'No-Show'], dtype=object)
```

```
[50]: data.dtypes
```

```
[50]: hotel                object
      is_canceled        int64
      lead_time          int64
      arrival_date_year  int64
      arrival_date_month object
      arrival_date_week_number int64
      arrival_date_day_of_month int64
      stays_in_weekend_nights int64
      stays_in_week_nights  int64
```

adults	int64
children	float64
babies	int64
meal	object
is_repeated_guest	int64
previous_cancellations	int64
previous_bookings_not_canceled	int64
reserved_room_type	object
assigned_room_type	object
booking_changes	int64
deposit_type	object
days_in_waiting_list	int64
customer_type	object
adr	float64
required_car_parking_spaces	int64
total_of_special_requests	int64
reservation_status	object
dtype:	object

```
[57]: from sklearn.preprocessing import LabelEncoder
```

```
[61]: hotel = LabelEncoder()
      nhotel = hotel.fit_transform(data[DhotelID])
      data[DhotelID] = nhotel
      data = data.astype({DhotelID:Dint64D})
      np.unique(nhotel)
```

```
[61]: array([0, 1], dtype=int64)
```

```
[62]: arrival_date_month = LabelEncoder()
      n_arrival_date_month = arrival_date_month.
      ↪fit_transform(data[Darrival_date_monthD])
      data[Darrival_date_monthD] = n_arrival_date_month
      data = data.astype({Darrival_date_monthD:Dint64D})
      np.unique(n_arrival_date_month)
```

```
[62]: array([ 0,  1,  2,  3,  4,  5,  6,  7,  8,  9,10,11])
```

```
[63]: deposit_type = LabelEncoder()
      n_deposit_type = hotel.fit_transform(data[Ddeposit_typeD])
      data[Ddeposit_typeD] = n_deposit_type
      data = data.astype({Ddeposit_typeD:Dint64D})
      np.unique(n_deposit_type)
```

```
[63]: array([0, 1, 2])
```

```
[64]: customer_type = LabelEncoder()
      n_customer_type = customer_type.fit_transform(data[Dcustomer_typeD])
      data[Dcustomer_typeD] = n_customer_type
      data = data.astype({Dcustomer_typeD:Dint64D})
      np.unique(n_customer_type)
```

```
[64]: array([0, 1, 2, 3])
```

```
[65]: reserved_room_type = LabelEncoder()
n_reserved_room_type = reserved_room_type.
↳fit_transform(data[Dreserved_room_typeD])
data[DReserved_room_typeD] = n_reserved_room_type
data = data.astype({DReserved_room_typeD:Dint64D})
np.unique(n_reserved_room_type)
```

[65]: array([0, 1, 2, 3, 4, 5, 6, 7, 8])

```
[66]: assigned_room_type = LabelEncoder()
n_assigned_room_type = assigned_room_type.
↳fit_transform(data[Dassigned_room_typeD])
data[Dassigned_room_typeD] = n_assigned_room_type
data = data.astype({Dassigned_room_typeD:Dint64D})
np.unique(n_assigned_room_type)
```

[66]: array([0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10])

```
[67]: meal = LabelEncoder()
nmeal = meal.fit_transform(data[DmealD])
data[DmealD] = nmeal
data = data.astype({DmealD:Dint64D})
np.unique(nmeal)
```

[67]: array([0, 1, 2, 3, 4])

```
[68]: reservation_status = LabelEncoder()
n_reservation_status = reservation_status.
↳fit_transform(data[Dreservation_statusD])
data[Dreservation_statusD] = n_reservation_status
data = data.astype({Dreservation_statusD:Dint64D})
np.unique(n_reservation_status)
```

[68]: array([0, 1, 2])

```
[69]: data.dtypes
```

```
[69]: hotel                                int64
is_canceled                             int64
lead_time                               int64
arrival_date_year                        int64
arrival_date_month                      int64
arrival_date_week_number                int64
arrival_date_day_of_month               int64
stays_in_weekend_nights                 int64
stays_in_week_nights                   int64
adults                                  int64
children                                float64
babies                                  int64
meal                                    int64
is_repeated_guest                       int64
previous_cancellations                  int64
previous_bookings_not_canceled          int64
reserved_room_type                      int64
```

assigned_room_type	int64
booking_changes	int64
deposit_type	int64
days_in_waiting_list	int64
customer_type	int64
adr	float64
required_car_parking_spaces	int64
total_of_special_requests	int64
reservation_status	int64
dtype:	object

5. Разделение выборки на обучающую и тестовую

```
[73]: from sklearn.model_selection import train_test_split
```

```
[74]: X_train, X_test, y_train, y_test = train_test_split(data, data.hotel,
    ↪random_state=1)
```

Размеры обучающей выборки и тестовой выборки:

```
[76]: X_train.shape, y_train.shape, X_test.shape, y_test.shape
```

```
[76]: ((89046, 26), (89046,), (29682, 26), (29682,))
```

6. Построение моделей классификации

7. Логистическая регрессия

Построим модель логистической регрессии с помощью LogisticRegression:

```
[80]: from sklearn.linear_model import LogisticRegression
```

```
[81]: model_logistic = LogisticRegression()
    model_logistic.fit(X_train, y_train)
```

```
[81]: LogisticRegression()
```

```
[82]: targ_logistic = model_logistic.predict(X_test)
```

8. Случайный лес

Построим модель случайного леса с помощью RandomForestClassifier с числом соседей =

5:

```
[83]: from sklearn.ensemble import RandomForestClassifier
```

```
[84]: model_forest = RandomForestClassifier(n_estimators=5, oob_score=True,
    ↪random_state=100)
    model_forest.fit(X_train, y_train)
```

```
[84]: RandomForestClassifier(n_estimators=5, oob_score=True, random_state=100)
```

```
[85]: targ_forest = model_forest.predict(X_test)
```

Оценка качества моделей

Для оценки качества моделей классификации будем использовать две метрики - Ассигасу и матрицу ошибок Confusion Matrix. Именно эти две метрики были выбраны, так как они помогают посчитать процент верно определенных отелей и отобразить наглядно классифицированные данные.

```
[86]: from sklearn.metrics import accuracy_score
from sklearn.metrics import confusion_matrix
from sklearn.metrics import ConfusionMatrixDisplay
```

9. Метрика Accuracy

```
[87]: accuracy_score(y_test, targ_logistic)
```

```
[87]: 0.7895020551175796
```

```
[88]: accuracy_score(y_test, targ_forest)
```

```
[88]: 1.0
```

Видно, что точность всех классов у модели случайного леса значительно выше, чем у модели логистической регрессии.

Проверим для каждого отеля из hotel:

```
[90]: from typing import Dict, Tuple
```

```
[92]: def accuracy_score_for_classes(y_true:
    np.ndarray,
    y_pred: np.ndarray) -> Dict[int, float]:
    """
    Вычисление метрики Ассигасу для каждого класса
    y_true - истинные значения классов
    y_pred - предсказанные значения классов
    Возвращает словарь: ключ - метка
    класса, значение - Ассигасу для данного
    класса """
    # Для удобства фильтрации сформируем Pandas DataFrame
    d = {'t': y_true, 'p': y_pred} df =
    pd.DataFrame(data=d)
    # Метки классов
    classes = np.unique(y_true) #
    Результирующий словарь res
    = dict()
    # Перебор меток классов
    for c in classes:
        # отфильтруем данные, которые
        соответствуют # текущей метке класса в
        истинных значениях temp_dataflt = df[df['t']==c]
        # расчет ассигасу для заданной метки класса
        temp_acc = accuracy_score(
            temp_dataflt['t'].values,
            temp_dataflt['p'].values)
```

```

        # сохранение результата в словарь
        res[c] = temp_acc
    return res

def print_accuracy_score_for_classes(
    y_true: np.ndarray,
    y_pred: np.ndarray):
    """
    Вывод метрики Accuracy для каждого класса
    """
    accs = accuracy_score_for_classes(y_true, y_pred)
    if len(accs) > 0:
        print('Отель \t Accuracy')
    for i in accs:
        print('{} \t {}'.format(i, accs[i]))

```

```
[93]: print_accuracy_score_for_classes(y_test, targ_logistic)
```

Отель	Accuracy
0	0.894289927289368
1	0.5837244133799301

```
[94]: print_accuracy_score_for_classes(y_test, targ_forest)
```

Отель	Accuracy
0	1.0
1	1.0

10. Метрика Confusion Matrix

```
[95]: confusion_matrix(y_test, targ_logistic, labels=[0, 1, 2, 3, 4, 5, 6, 7])
```

```
[95]: array([[17588, 2079, 0, 0, 0, 0, 0, 0],
            [ 4169, 5846, 0, 0, 0, 0, 0, 0],
            [    0,    0, 0, 0, 0, 0, 0, 0],
            [    0,    0, 0, 0, 0, 0, 0, 0],
            [    0,    0, 0, 0, 0, 0, 0, 0],
            [    0,    0, 0, 0, 0, 0, 0, 0],
            [    0,    0, 0, 0, 0, 0, 0, 0],
            [    0,    0, 0, 0, 0, 0, 0, 0]],
      dtype=int64)
```

```
[96]: confusion_matrix(y_test, targ_forest, labels=[0, 1, 2, 3, 4, 5, 6, 7])
```

```
[96]: array([[19667, 0, 0, 0, 0, 0, 0, 0],
            [    0, 10015, 0, 0, 0, 0, 0, 0],
            [    0,    0, 0, 0, 0, 0, 0, 0],
            [    0,    0, 0, 0, 0, 0, 0, 0],
            [    0,    0, 0, 0, 0, 0, 0, 0],
            [    0,    0, 0, 0, 0, 0, 0, 0],
            [    0,    0, 0, 0, 0, 0, 0, 0],
            [    0,    0, 0, 0, 0, 0, 0, 0]],
      dtype=int64)
```

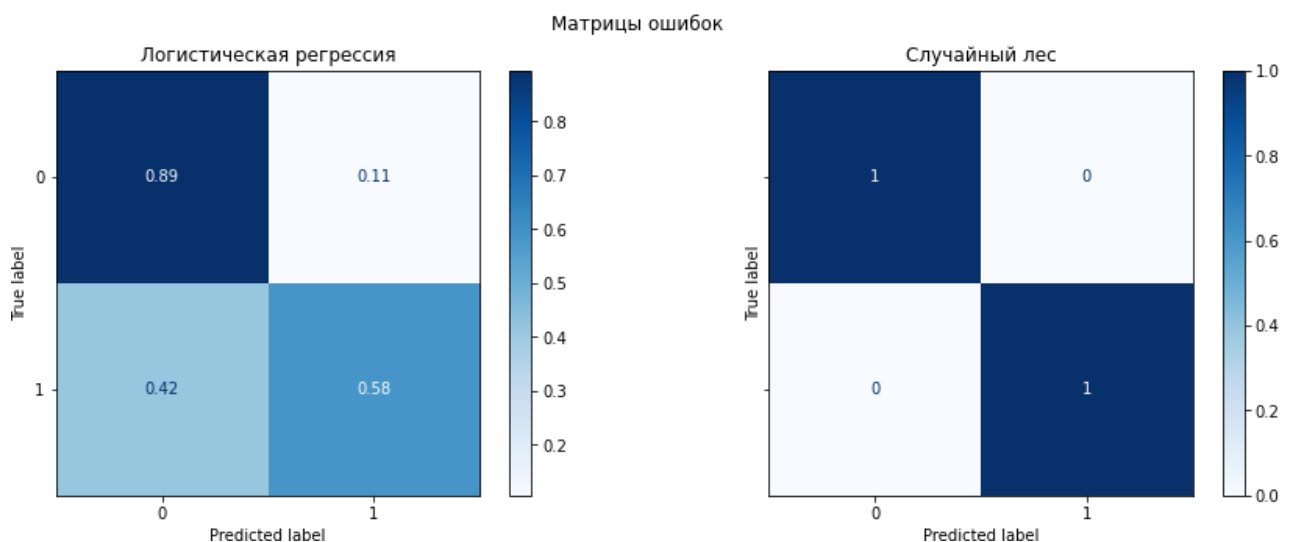
```
[ 0, 0, 0, 0, 0, 0, 0, 0],
dtype=int64)
```

```
[97]: fig, ax = plt.subplots(1, 2, sharex='col', sharey='row', figsize=(15,5))
```

```
ConfusionMatrixDisplay.from_estimator(
    model_logistic,
    X_test,
    y_test,
    display_labels=model_logistic.classes_,
    cmap=plt.cm.Blues,
    normalize='true',
    ax=ax[0]
)
```

```
ConfusionMatrixDisplay.from_estimator(
    model_forest,
    X_test,
    y_test,
    display_labels=model_forest.classes_,
    cmap=plt.cm.Blues,
    normalize='true',
    ax=ax[1]
)
```

```
fig.suptitle('Матрицы ошибок')
ax[0].title.set_text('Логистическая регрессия')
ax[1].title.set_text('Случайный лес')
```



Видно, что модель случайного леса обладает высокой точностью.