GeekBrains, ML in Business

Lesson 4 Homework

Ссылки:

- https://towardsdatascience.com/a-quick-uplift-modeling-introduction-6e14de32bfe0
- https://habr.com/ru/company/ru_mts/blog/485980/#reference1
- https://en.wikipedia.org/wiki/Uplift_modelling
- https://www.youtube.com/watch?v=yFQAIJBYXI0
- https://www.youtube.com/watch?v=jCUcYiBK03I
- https://www.uplift-modeling.com/en/latest/
- https://arxiv.org/pdf/1809.04559.pdf
- https://catboost.ai/docs/concepts/about.html

Библиотеки и пакеты:

- causalml
- scikit-uplift (sklift)
- catboost

Импорт библиотек

In [1]:
```
import numpy as np
import pandas as pd

from IPython.display import Image

from sklearn.model_selection import train_test_split

from sklift.metrics import uplift_at_k
from sklift.viz import plot_uplift_preds
from sklift.models import SoloModel, ClassTransformation, TwoModels

from causalml.inference.tree import UpliftTreeClassifier, UpliftRandomForestClassifier
from causalml.inference.tree import uplift_tree_string, uplift_tree_plot

from catboost import CatBoostClassifier


# %matplotlib inline
```

Задание 1

Скачать набор данных маркетинговых кампаний отсюда https://www.kaggle.com/davinwijaya/customer-retention.

Решение Задания 1

Скачали данные, импортируем.

In [2]:
```
df = pd.read_csv('data.csv', delimiter=',')
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 64000 entries, 0 to 63999
Data columns (total 9 columns):
 #   Column         Non-Null Count  Dtype  
---  ------         --------------  -----  
 0   recency        64000 non-null  int64  
 1   history        64000 non-null  float64
 2   used_discount  64000 non-null  int64  
 3   used_bogo      64000 non-null  int64  
 4   zip_code       64000 non-null  object 
 5   is_referral    64000 non-null  int64  
 6   channel        64000 non-null  object 
 7   offer          64000 non-null  object 
 8   conversion     64000 non-null  int64  
dtypes: float64(1), int64(5), object(3)
memory usage: 4.4+ MB
```

In [3]:
```
df.head()
```

Out[3]:

	recency	history	used_discount	used_bogo	zip_code	is_referral	channel	offer	conversion
0	10	142.44	1	0	Surburban	0	Phone	Buy One Get One	0
1	6	329.08	1	1	Rural	1	Web	No Offer	0
2	7	180.65	0	1	Surburban	1	Web	Buy One Get One	0
3	9	675.83	1	0	Rural	1	Web	Discount	0

	recency	history	used_discount	used_bogo	zip_code	is_referral	channel	offer	conversion
4	2	45.34	1	0	Urban	0	Web	Buy One Get One	0

In [4]:
```
df['conversion'].value_counts()
```

Out[4]:
```
0    54606
1     9394
Name: conversion, dtype: int64
```

In [5]:
```
df['offer'].value_counts()
```

Out[5]:
```
Buy One Get One    21387
Discount           21307
No Offer           21306
Name: offer, dtype: int64
```

Задание 2

Поле conversion - это целевая переменная, а offer - коммуникация. Переименовать поля (conversion -> target, offer -> treatment) и привести поле treatment к бинарному виду (1 или 0, т.е было какое-то предложение или нет) - значение No Offer означает отсутствие коммуникации, а все остальные - наличие.

Решение Задания 2

Переименовываем.

In [6]:
```
df = df.rename(columns={'conversion': 'target', 'offer': 'treatment'})
```

Приводим в бинарный вид и удаляем признак treatment_No Offer - он не дает никакой дополнительной информации, т.к. нули в 2-х других признаках автоматически означают No Offer.

In [7]:
```
df.loc[df['treatment'] != 'No Offer', 'treatment'] = 1
df.loc[df['treatment'] == 'No Offer', 'treatment'] = 0
df['treatment'] = df['treatment'].astype(np.uint8)
```

In [8]:
```
df.head()
```

Out[8]:

	recency	history	used_discount	used_bogo	zip_code	is_referral	channel	treatment	target
0	10	142.44	1	0	Surburban	0	Phone	1	0
1	6	329.08	1	1	Rural	1	Web	0	0
2	7	180.65	0	1	Surburban	1	Web	1	0
3	9	675.83	1	0	Rural	1	Web	1	0
4	2	45.34	1	0	Urban	0	Web	1	0

Задание 3

Сделать разбиение набора данных не тренировочную и тестовую выборки.

Решение Задания 3

Делим датасет.

In [9]:
```
X_train, X_test, y_train, y_test, treat_train, treat_test = train_test_split(df.drop(columns=['target']),
                                                            df['target'],
                                                            df['treatment'],
                                                            random_state=0)
```

In [10]:
```
X_train.shape, X_test.shape, y_train.shape, y_test.shape, treat_train.shape
```

Out[10]:
```
((48000, 8), (16000, 8), (48000,), (16000,), (48000,))
```

Задание 4

Сделать feature engineering на ваше усмотрение (допускается свобода выбора методов).

Решение Задания 4

Выделим бины пользователей по сумме их покупок. Выделим китов, дельфинов и пескарей.

In [11]:
```
df['history'].describe()
```

Out[11]:
```
count    64000.000000
mean       242.085656
std        256.158608
min         29.990000
25%         64.660000
50%        158.110000
75%        325.657500
max       3345.930000
Name: history, dtype: float64
```

Пусть китами будут пользователи с покупками на сумму >= 1000, дельфинами - на сумму 200-1000, пескари - < 200.

In [12]:
```
def segment_customers(df: pd.DataFrame) -> pd.DataFrame:
    df = df.copy()
    df['customer_is_whale'] = 0
    df['customer_is_dolphin'] = 0
    df['customer_is_minnow'] = 0
    df.loc[df['history'] >= 1000, 'customer_is_whale'] = 1
    df.loc[(df['history'] < 1000) & (df['history'] >= 200), 'customer_is_dolphin'] = 1
    df.loc[df['history'] < 200, 'customer_is_minnow'] = 1
    return df
```

In [13]:
```
X_train = segment_customers(X_train)
X_test = segment_customers(X_test)
```

In [14]:
```
X_train.head()
```

Out[14]:

	recency	history	used_discount	used_bogo	zip_code	is_referral	channel	treatment	customer_is_whale	customer_is_dolphin	customer_is_minno
1098	8	63.58	1	0	Surburban	1	Phone	1	0	0	
13764	3	395.35	1	0	Surburban	1	Web	1	0	1	
45116	4	1307.99	1	1	Rural	1	Phone	1	1	0	
15363	10	159.01	0	1	Surburban	1	Web	0	0	0	
44498	1	276.00	1	0	Urban	0	Phone	0	0	1	

Задание 5

Провести uplift-моделирование 3 способами: одна модель с признаком коммуникации (S learner), модель с трансформацией таргета (трансформация классов п. 2. 1) и вариант с двумя независимыми моделями.

Решение Задания 5

Датафрейм для сравнения метрик.

In [15]:
```
metrics_df = pd.DataFrame(columns=['uplift@10%', 'uplift@20%'])
```

Категориальные признаки.

In [16]:
```
cat_features = ['zip_code', 'channel']
```

Solo Learner

In [17]:
```
sm = SoloModel(CatBoostClassifier(iterations=20, thread_count=2, random_state=42, silent=True))
sm = sm.fit(X_train, y_train, treat_train, estimator_fit_params={'cat_features': cat_features})

uplift_sm = sm.predict(X_test)
print(uplift_sm)

sm_score_10 = uplift_at_k(y_true=y_test, uplift=uplift_sm, treatment=treat_test, strategy='by_group', k=0.1)
sm_score_20 = uplift_at_k(y_true=y_test, uplift=uplift_sm, treatment=treat_test, strategy='by_group', k=0.2)
```

```
[0.06326457 0.06920032 0.07275016 ... 0.06685124 0.02610939 0.07757569]
```

Сохраняем метрики модели.

In [18]:
```
metrics_df = metrics_df.append({'uplift@10%': sm_score_10, 'uplift@20%': sm_score_20}, ignore_index=True)
```

Transform Learner

In [19]:
```
ct = ClassTransformation(CatBoostClassifier(iterations=20, thread_count=2, random_state=42, silent=True))
ct = ct.fit(X_train, y_train, treat_train, estimator_fit_params={'cat_features': cat_features})

uplift_ct = ct.predict(X_test)
print(uplift_ct)

ct_score_10 = uplift_at_k(y_true=y_test, uplift=uplift_ct, treatment=treat_test, strategy='by_group', k=0.1)
ct_score_20 = uplift_at_k(y_true=y_test, uplift=uplift_ct, treatment=treat_test, strategy='by_group', k=0.2)
```

```
It is recommended to use this approach on treatment balanced data. Current sample size is unbalanced.
```

```
[-0.68133514 -0.66582944 -0.61137839 ...  0.78141424 -0.65206333
 -0.32893603]
```

Сохраняем метрики модели.

In [20]:
```
metrics_df = metrics_df.append({'uplift@10%': ct_score_10, 'uplift@20%': ct_score_20}, ignore_index=True)
```

Two Model Learner

In [21]:
```
tm = TwoModels(
    estimator_trmnt=CatBoostClassifier(iterations=20, thread_count=2, random_state=42, silent=True),
    estimator_ctrl=CatBoostClassifier(iterations=20, thread_count=2, random_state=42, silent=True),
    method='vanilla'
)
tm = tm.fit(
    X_train, y_train, treat_train,
    estimator_trmnt_fit_params={'cat_features': cat_features},
    estimator_ctrl_fit_params={'cat_features': cat_features}
)

uplift_tm = tm.predict(X_test)
print(uplift_tm)

tm_score_10 = uplift_at_k(y_true=y_test, uplift=uplift_tm, treatment=treat_test, strategy='by_group', k=0.1)
tm_score_20 = uplift_at_k(y_true=y_test, uplift=uplift_tm, treatment=treat_test, strategy='by_group', k=0.2)
```

```
[0.08574949 0.08029174 0.09133908 ... 0.07039956 0.0432995  0.12032493]
```

Сохраняем метрики модели.

In [22]:
```
metrics_df = metrics_df.append({'uplift@10%': tm_score_10, 'uplift@20%': tm_score_20}, ignore_index=True)
```

Задание 6

В конце вывести единую таблицу сравнения метрик uplift@10%, uplift@20% этих 3 моделей.

Решение Задания 6

In [23]:
```
metrics_df
```

Out[23]:

	uplift@10%	uplift@20%
0	0.059370	0.068407
1	0.225533	0.195837
2	0.079349	0.065813

Лучше всего себя показала модель с трансформацией классов.

Задание 7

Построить модель UpliftTreeClassifier и попытаться описать словами полученное дерево.

Решение Задания 7

In [24]:
```
def transform_data_to_tree(df: pd.DataFrame) -> pd.DataFrame:
    df = df.copy()
    df = pd.get_dummies(df, drop_first=True)
    return df
```

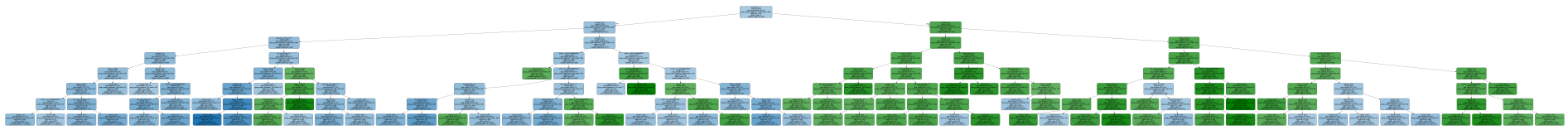
In [25]:
```
X_train_tree = transform_data_to_tree(X_train)
X_test_tree = transform_data_to_tree(X_test)
features = [col for col in X_train_tree]
```

In [27]:
```
uplift_model = UpliftTreeClassifier(max_depth=8, min_samples_leaf=200, min_samples_treatment=50,
                                    n_reg=100, evaluationFunction='KL', control_name='control')

uplift_model.fit(X_train_tree.values,
                 treatment=treat_train.map({1: 'treatment1', 0: 'control'}).values,
                 y=y_train)

graph = uplift_tree_plot(uplift_model.fitted_uplift_tree, features)
Image(graph.create_png())
```

Out[27]:



Можно выделить некоторые сегменты:

- Городские жители, давно не покупавшие и использовавшие предложение Buy One Get One с суммой покупок 322-388 - для них веротность конверсии ухудшится при коммуникации,
- Клиенты, использовавшие предложение BOGO, которые недавно покупали у нас и имеют общую сумму покупок < 52.5 - для них коммуникация повысит шанс конверсии,
- Пригородные жители не по рефералам, покупающие по телефону, с общей суммой покупок > 200 - для них коммуникация повысит шанс конверсии.

Задание 8*

Для модели S learner (модель с дополнительным признаком коммуникации) построить зависимость таргета (конверсии - поле conversion) от значения uplift: 1) сделать прогноз и получить uplift для тестовой выборки 2) отсортировать тестовую выборку по uplift по убыванию 3) разбить на децили (pandas qcut вам в помощь) 4) для каждого дециля посчитать среднюю conversion.

Решение Задания 8*

In [ ]:

Задание 9*

Построить модель UpliftRandomForestClassifier и попытаться описать словами полученное дерево.

Решение Задания 9*

In [ ]: