

Glow Byte Hackaton 2021



Команда “Mental”
Кейс 2

Задача: усовершенствовать систему
скоринга кредитных заявок физических
лиц путем добавления к анкете данных из
других источников

Система scoring сейчас



Дорожная карта продукта команды “Mental”

Новые данные

Выбор данных из разных баз для дополнения анкеты

1

Автоматизация

Написание скриптов для автоматического подключения к базам и выгрузки данных

2

Подготовка

Фильтрация, обработка и подготовка данных к использованию

3

Объединение

Добавление данных к анкете

Машинное обучение

Выявление параметров, значимых для кредитоспособности, из объединенного массива данных

4

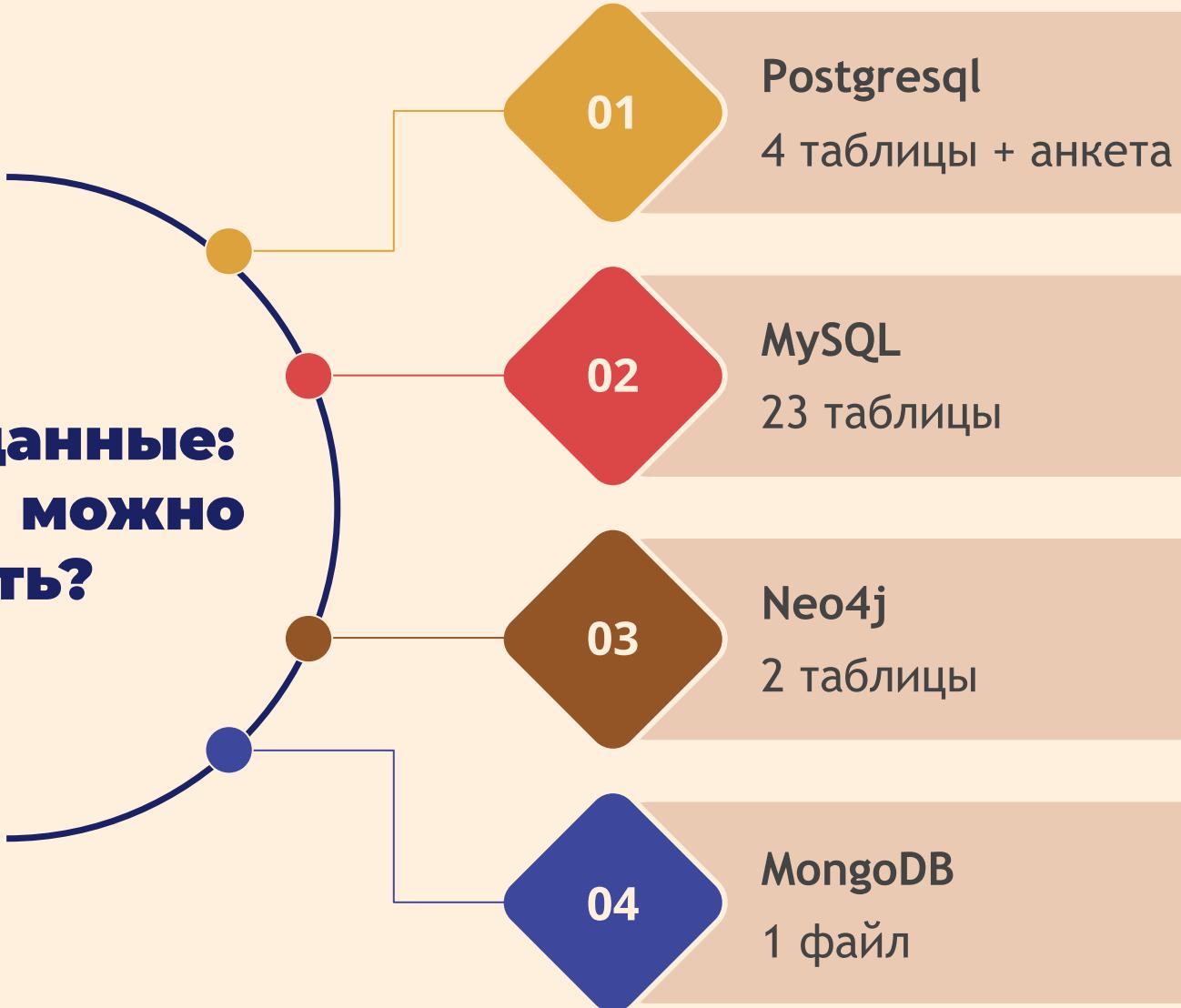
5

Выбор новых данных для дополнения анкеты

Какие данные уже есть и какие из них нужны?

Общая характеристика данных	БД
Данные о кредитных продуктах	postgresql
Данные о зарплатных продуктах	neo4j
Данные о страховых продуктах	mysql
Транскрипты интервью	mongodb

Выгруженные данные: много, но все ли можно использовать?



Первичный анализ данных на релевантность для дополнения анкеты

Postgresql: из 4 таблиц можно использовать 2

MySQL: из 23 таблиц выбрано 2 для объединения с анкетой

Neo4j: можно использовать 1 по связям с зарплатным проектом

Mongodb: данные содержат серьезные ошибки и до использования должны быть исправлены и отфильтрованы

Причины отсеивания: таблицы не содержат уникальные ID, уже повторяют данные анкеты, содержат ошибки, не являются релевантными для конкретной задачи

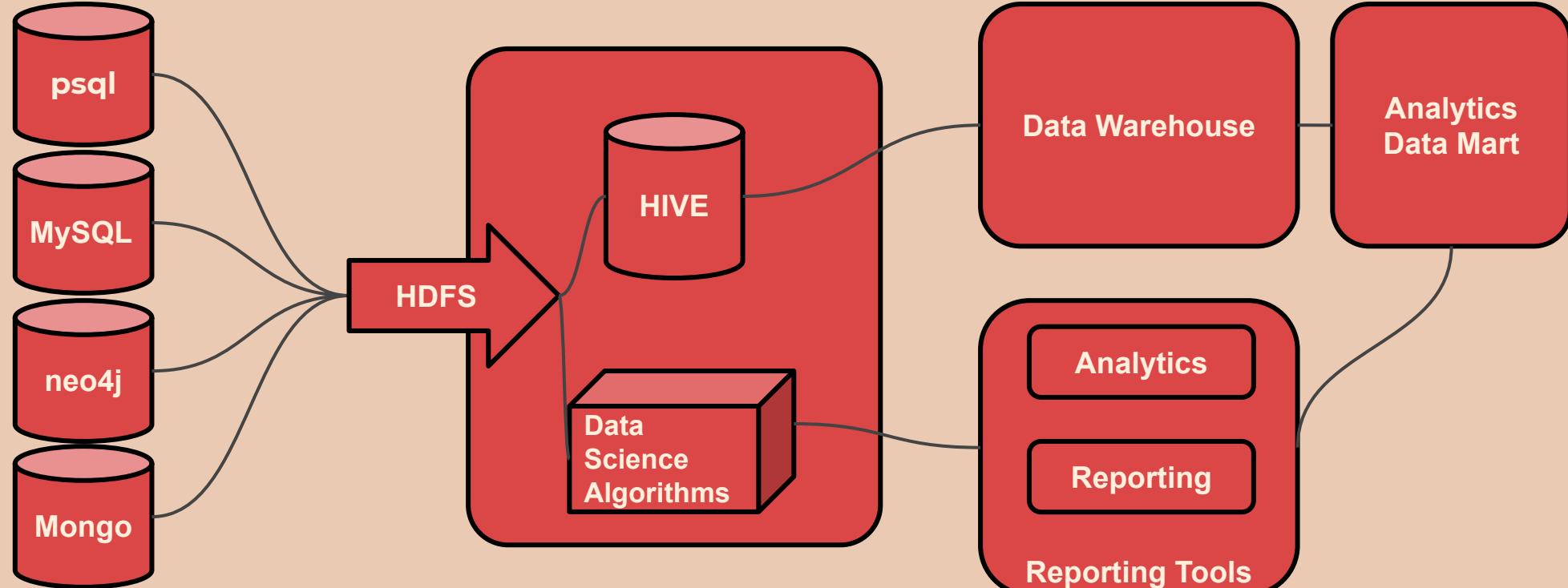
Пример ошибки в транскрипте

I7: 'Чем вы занимаетесь',
A7: 'Я таксист',
I8: 'Хорошо, у вас в какой форме вы устроены?',
A8: 'Пятьдесят шестьдесят тысяч рублей обычно',

	LOAN_ID	LOAN_BALANCE	OD_AMT	INT_AMT	OD_OVERBUE_AMT	INT_OVERBUE_AMT	START_DT	END_DT
0	15556671000	3945600	267040.0	0.0	0.0	2021-08-20 00:00:00	2021-09-04 00:00:00	NaN
1	15556671000	3942700	261540.0	0.0	0.0	2021-09-05 00:00:00	2999-01-01 00:00:00	NaN
2	15556671001	357800	24780.0	3670.0	550.0	2021-08-20 00:00:00	2999-01-01 00:00:00	NaN
3	15556671002	134500	22760.0	0.0	0.0	2021-08-05 00:00:00	2021-09-09 00:00:00	NaN
4	15556671002	134500	25678.0	0.0	0.0	2021-09-10 00:00:00	2021-09-11 00:00:00	NaN
5	15556671002	128650	20653.0	0.0	0.0	2021-09-12 00:00:00	2999-01-01 00:00:00	NaN
6	15556671003	0	0.0	0.0	0.0	2021-08-05 00:00:00	2999-01-01 00:00:00	NaN
7	15556671004	0	0.0	189034.0	36905.0	2021-07-31 00:00:00	2999-01-01 00:00:00	NaN
8	15556671005	0	0.0	0.0	0.0	2021-06-30 00:00:00	2999-01-01 00:00:00	NaN

Пример отсутствия уникальных ID

Данные выгружены. Где и как хранить?



Написание скриптов для автоматического подключения к базам и выгрузки данных

Почему автоматизация?

Возможность проводить изменения системы скоринга регулярно + обязанность банка хранить данные, на основании которых были приняты решения о кредитовании.

Данные в базах систематически обновляются и исправляются. Работа непосредственно с базой данных должна повторяться каждый раз при обновлении расчета скоринга, но данные, на основании которых были сделаны ранние расчеты, должны храниться.

Однако, если баз много, это занимает много времени и для его экономии необходимы автоматические скрипты.

```
root@gbc-VBox:/var/lib/mysql-files# ls | grep txt
CLIENT_BIRTH_DT.txt
CLIENT_CITIZENSHIP.txt
CLIENT_FIO.txt
CLIENT_INN.txt
CLIENT.txt
CONTRACT_CLOSE_DT.txt
CONTRACT_INSURANCE_AMT.txt
CONTRACT_INSURANCE_RATE.txt
CONTRACT_NUM.txt
CONTRACT_OPEN_DT.txt
INSURANCE_PRODUCT_2_GROUP.txt
INSURANCE_PRODUCT_3_GROUP.txt
INSURANCE_PRODUCT_NET_RATE.txt
INSURANCE_PRODUCT_RISK.txt
INSURANCE_PRODUCT.txt
ISUTANCE_CONTRACT.txt
ISUTANCE_CONTRACT_X_CLIENT.txt
ISUTANCE_CONTRACT_X_PLEDGE.txt
ISUTANCE_CONTRACT_X_PRODUCT.txt
PLEDGE_DISCONT.txt
PLEDGE_INIT_VALUE_AMT.txt
PLEDGE.txt
PLEDGE_TYPE.txt
root@gbc-VBox:/var/lib/mysql-files#
```



Упрощенно: от такого процесса планируется уйти к автоматизации

Решение реализуется на языке Python3. Примеры автоматизации.

```
import psycopg2

con = psycopg2.connect(
    database="postgres",
    user="postgres",
    password="Kaliakakya",
    host="127.0.0.1",
    port="5432"
)

print("Database opened successfully")
cur = con.cursor()
cur.execute("SELECT admission, name, age, course,
department from STUDENT")

rows = cur.fetchall()
for row in rows:
    print("ADMISSION =", row[0])
    print("NAME =", row[1])
    print("AGE =", row[2])
    print("COURSE =", row[3])
    print("DEPARTMENT =", row[4], "\n")

print("Operation done successfully")

con.close()
```



PostgreSQL : библиотека psycopg2

```
pip install py2neo
from py2neo import Graph, Node, Relationship
graph = Graph("localhost:7474", auth=("neo4j", "gbc_user"))
apoc.export.file.enabled=true
CALL apoc.export.csv.all("neo4j_data.csv", {})
CALL apoc.export.csv.all(null, {stream:true})
YIELD file, nodes, relationships, properties, data
RETURN file, nodes, relationships, properties, data
```



: библиотека py2neo,
Cypher query language и
APOC library

```
client = pymongo.MongoClient("mongodb://localhost:27017/")
db = client["database_name"]
col = db["collection_name"]
x = col.find_one()
print(x)
```



mongoDB : библиотека Pymongo

Фильтрация, обработка и подготовка данных к использованию

Решение реализуется на языке Python3 при помощи Pandas

Необходимо:

- Проверить, какие данные можно убрать из анкеты
- Данные полученных таблиц совместить с заголовками из документа Excel с описаниями БД
- Файлы нетабличного характера распарсить (разделить на отдельные единицы информации в таблицах), и также совместить с заголовками
- Отфильтровать данные, по которым нельзя ориентироваться
- Добавить данные к анкете (по наличию общего ID)

Обработка анкеты (Postgresql)

```
pg_apps = pd.read_csv("postgres_prod_apps.csv", header=None, names=[ "APP_ID", "APP_DATE", "CRED_AMOUNT", "CRED_TERM",
"CRED_OBJECT", "CUST_MONTH_INCOME",
"CUST_FAMILY_MONTH_INCOME",
"CUST_FIO", "CUST_ID", "CUST_BIRTH", "CUST_INN",
"GUARANTOR_FLAG",
"GUARANTOR_FIO", "GUARANTOR_PHONE", "GUARANTOR_BIRTH",
"PLEDGE_AMOUNT", "PLEDGE_TYPE", "INTERVIEW_DATETIME",
"CUST_RELATION_BANK_TYPE", "APP_SALE_CHANNEL"])

pg_apps = pg_apps.fillna(0)
pg_apps = pg_apps.drop(columns={"APP_ID", "APP_DATE", "CUST_FIO", "CUST_INN", "GUARANTOR_FIO", "GUARANTOR_PHONE",
"PLEDGE_TYPE", "INTERVIEW_DATETIME"})
pg_apps[ 'CUST_BIRTH' ] = pd.DatetimeIndex(pg_apps[ 'CUST_BIRTH' ]).year
pg_apps[ "GUARANTOR_BIRTH" ]=pd.DatetimeIndex(pg_apps[ "GUARANTOR_BIRTH" ]).year
pg_apps
```

Было 20 колонок, осталось 12

Подготовка анкеты к анализу и добавлению данных: добавление названий колонок, удаление нерелевантных колонок для анализа, перевод даты рождения в год рождения, замена отсутствующих значений на 0

	CRED_AMOUNT	CRED_TERM	CRED_OBJECT	CUST_MONTH_INCOME	CUST_FAMILY_MONTH_INCOME	CUST_ID	CUST_BIRTH	GUARANTOR_FLAG	GUARANTOR_BIRTH	PLEDGE_AMOUNT	CUST_RELATION_BANK_TYPE	APP_SALE_CHANNEL
0	100500.0	5	Выдача средств	35000.0	75000.0	100200300	1982	f		1970	0.0	0
1	200000.0	12	Кредит на обучение	0.0	0.0	100200301	1984	t		1984	0.0	0
2	55000.0	6	Выдача средств	0.0	0.0	100200302	1966	f		1970	0.0	0
3	15040000.0	25	Городская недвижимость	435000.0	0.0	100200303	2001	t		1999	21509000.0	Инсайдер
4	654600.0	15	Оборудование	0.0	0.0	100200304	1968	t		1983	150000.0	0
5	2267000.0	20	Авто	120.0	157.0	100200305	1990	t		1984	3200000.0	0
6	559000.0	6	Земельный участок	66.0	150.0	100200306	1984	f		1970	680000.0	0
7	1547000.0	15	Авто	160.0	0.0	100200307	1949	t		1991	1599000.0	0
8	8838925.0	36	Дача	95.0	143.0	100200308	1984	t		1988	0.0	Сотрудник банка
9	502000.0	20	Выдача средств	0.0	0.0	100200309	1988	f		1970	0.0	0
10	35500.0	5	Бытовая техника	0.0	0.0	100200310	2000	f		1970	0.0	0
11	987400.0	24	Авто	65.0	108.0	100200311	2002	t		1965	0.0	0
12	2502000.0	35	Коммерческая недвижимость	210.0	210.0	100200312	1999	t		1970	2830000.0	0

Процесс фильтрации и анализа данных : Postgresql

```
pg_lloans = pg_lloans.drop(columns={"PRODUCT_ID"})
pg_lloans[ 'BEGIN_DT' ] = pd.DatetimeIndex(pg_lloans[ 'BEGIN_DT' ]).year
pg_lloans[ "CLOSE_PLAN_DT" ]=pd.DatetimeIndex(pg_lloans[ "CLOSE_PLAN_DT" ]).year
pg_lloans[ "CLOSE_FACT_DT" ]=pd.DatetimeIndex(pg_lloans[ "CLOSE_FACT_DT" ]).year
pg_lloans = pg_lloans.fillna(0)
pg_lloans.CLOSE_FACT_DT = pg_lloans.CLOSE_FACT_DT.astype(int)
pg_lloans
```

	LOAN_ID	BEGIN_DT	CLOSE_PLAN_DT	CLOSE_FACT_DT	CLIENT_ID	REPAYMENT_MODE	INTEREST_RATE
0	15556671000	2019	2029	0	100200300	аннуитет	12.5
1	15556671001	2021	2023	0	100200302	дисконт	7.8
2	15556671002	2021	2022	0	100200304	аннуитет	11.3
3	15556671003	2020	2022	2021	100200306	аннуитет	10.0
4	15556671004	2016	2021	0	100200308	аннуитет	9.4
5	15556671005	2020	2021	2021	100200310	аннуитет	6.9

Пример подготовки
данных к объединению
(таблица с данными о
кредитах)

Процесс фильтрации и анализа данных : Postgresql

CLIENT_ID	FIO	BIRTH_DT	INN	CITIZENSHIP
0	100200300	Лопатин Иван Иванович ...	1982-03-02	589463240112
1	100200302	Захарова Людмила Айдаровна ...	1966-09-04	554567899997 Республика Беларусь
2	100200304	Романова Валентина Александровна ...	1968-10-16	234568923474
3	100200306	Сорокин Павел Леонидович ...	1984-10-17	345233350097
4	100200308	Сабиров Рамиль Маратович ...	1984-12-19	234525680984
5	100200310	Николаев Егор Ибрагимович ...	2000-01-21	398088777766

```
pg_clients = pg_clients.drop(columns=["FIO", "BIRTH_DT", "INN"])
pg_clients
```

CLIENT_ID	CITIZENSHIP
0	100200300
1	100200302 Республика Беларусь
2	100200304
3	100200306
4	100200308
5	100200310

Таблица повторяет данные из анкеты, за исключением гражданства (его можно добавить к анкетным данным)

Экстракция и обработка данных из Neo4j

neo4j_relations		
n	r	m
{"TURN_MONTH_AMT":75324,"CARD_ID":4555300019552003,"SALARY_MONTH_INCOME":45000}	0	{"SALARY_PROJECT_ID":12388933}
{"TURN_MONTH_AMT":192394,"CARD_ID":5478223498731265,"SALARY_MONTH_INCOME":145000}	0	{"SALARY_PROJECT_ID":12388933}
{"LEGAL_ENTITY_ID":100300315,"CLIENT_NAME":"МГТУ ДВКТ"}	0	{"SALARY_PROJECT_ID":12388933}
{"TURN_MONTH_AMT":35986,"CARD_ID":2093435614431892,"SALARY_MONTH_INCOME":2500}	0	{"SALARY_PROJECT_ID":123834355}
{"TURN_MONTH_AMT":1750,"CARD_ID":4578430918943344,"SALARY_MONTH_INCOME":1750}	0	{"SALARY_PROJECT_ID":123834355}
{"LEGAL_ENTITY_ID":100300315,"CLIENT_NAME":"МГТУ ДВКТ"}	0	{"SALARY_PROJECT_ID":123834355}
{"CLIENT_ID":100200301,"CLIENT_FIO":"Алексеев Борис Петрович"}	0	{"TURN_MONTH_AMT":192394,"CARD_ID":5478223498731265,"SALARY_MONTH_INCOME":145000}
{"CLIENT_ID":100200303,"CLIENT_FIO":"Макарова Елизавета Сергеевна"}	0	{"TURN_MONTH_AMT":75324,"CARD_ID":4555300019552003,"SALARY_MONTH_INCOME":45000}
{"CLIENT_ID":100200303,"CLIENT_FIO":"Макарова Елизавета Сергеевна"}	0	{"TURN_MONTH_AMT":1750,"CARD_ID":4578430918943344,"SALARY_MONTH_INCOME":1750}
{"CLIENT_ID":100200311,"CLIENT_FIO":"Лебедев Марк Генадьевич"}	0	{"TURN_MONTH_AMT":35986,"CARD_ID":2093435614431892,"SALARY_MONTH_INCOME":2500}
{"TURN_MONTH_AMT":35986,"CARD_ID":2093435614431892,"SALARY_MONTH_INCOME":2500}	0	{"PRODUCT_ID":111111}
{"TURN_MONTH_AMT":75324,"CARD_ID":4555300019552003,"SALARY_MONTH_INCOME":45000}	0	{"PRODUCT_ID":111111}
{"TURN_MONTH_AMT":1750,"CARD_ID":4578430918943344,"SALARY_MONTH_INCOME":1750}	0	{"PRODUCT_ID":111112}
{"TURN_MONTH_AMT":192394,"CARD_ID":5478223498731265,"SALARY_MONTH_INCOME":145000}	0	{"PRODUCT_ID":111113}
{"SALARY_PROJECT_ID":12388933}	0	{"PRODUCT_ID":111114}
{"SALARY_PROJECT_ID":12388933}	0	{"PRODUCT_ID":111115}
{"PRODUCT_1_GROUP_NM":Зарплатный проект}	0	{"PRODUCT_2_GROUP_NM":Зарплатный проект "Гибкость"}
{"PRODUCT_1_GROUP_NM":Дебетовая карта}	0	{"PRODUCT_2_GROUP_NM":Зарплатная дебетовая карта}
{"PRODUCT_ID":111114}	0	{"PRODUCT_3_GROUP_NM":ЗРПП "Гибкость" для сотрудников на удаленке}
{"PRODUCT_2_GROUP_NM":Зарплатный проект "Гибкость"}	0	{"PRODUCT_3_GROUP_NM":ЗРПП "Гибкость" для сотрудников на удаленке}
{"PRODUCT_ID":111115}	0	{"PRODUCT_3_GROUP_NM":ЗРПП "Гибкость" для студентов}
{"PRODUCT_2_GROUP_NM":Зарплатный проект "Гибкость"}	0	{"PRODUCT_3_GROUP_NM":ЗРПП "Гибкость" для студентов}
{"PRODUCT_2_GROUP_NM":Зарплатная дебетовая карта}	0	{"PRODUCT_3_GROUP_NM":Зарплатная карта "1000 мелочей"}
{"PRODUCT_ID":111112}	0	{"PRODUCT_3_GROUP_NM":Зарплатная карта "1000 мелочей"}
{"PRODUCT_ID":111113}	0	{"PRODUCT_3_GROUP_NM":Зарплатная карта "1000 мелочей"}
{"PRODUCT_2_GROUP_NM":Зарплатная дебетовая карта}	0	{"PRODUCT_3_GROUP_NM":Студенческая зарплатная карта}
{"PRODUCT_ID":111111}	0	{"PRODUCT_3_GROUP_NM":Студенческая зарплатная карта}



n	r	m
6	CLIENT_ID:100200301,CLIENT_FIO:Алексеев Борис Петрович	TURN_MONTH_AMT:192394,CARD_ID:5478223498731265,SALARY_MONTH_INCOME:145000
7	CLIENT_ID:100200303,CLIENT_FIO:Макарова Елизавета Сергеевна	TURN_MONTH_AMT:75324,CARD_ID:4555300019552003,SALARY_MONTH_INCOME:45000
8	CLIENT_ID:100200303,CLIENT_FIO:Макарова Елизавета Сергеевна	TURN_MONTH_AMT:1750,CARD_ID:4578430918943344,SALARY_MONTH_INCOME:1750
9	CLIENT_ID:100200311,CLIENT_FIO:Лебедев Марк Генадьевич	TURN_MONTH_AMT:35986,CARD_ID:2093435614431892,SALARY_MONTH_INCOME:2500

Обработка данных из Neo4j

```
names = client_salary['n'].str.split(", ", expand=True)
money = client_salary['m'].str.split(", ", expand=True)
client_salary = pd.concat([names,money], axis=1, ignore_index=True)
client_salary.reset_index(drop=True, inplace=True)
client_salary = client_salary.drop(columns={1,3})
client_salary = client_salary.rename(columns={client_salary.columns[0]:"CLIENT_ID",
                                              client_salary.columns[1]:"TURN_MONTH_AMT",
                                              4:"SALARY_MONTH_INCOME"})
client_salary = client_salary.replace({'CLIENT_ID':' ','TURN_MONTH_AMT':' ','SALARY_MONTH_INCOME':' '}, regex=True)
client_salary
```

	CLIENT_ID	TURN_MONTH_AMT	SALARY_MONTH_INCOME
0	100200301	192394	145000
1	100200303	75324	45000
2	100200303	1750	1750
3	100200311	35986	2500

Итоговая подвыборка данных о зарплате, готовая для объединения

Mongodb: обработка текста

После исправления ошибок в базе можно обратиться к ней за недостающими данными для обработки дополненной анкеты.

Например, это проверка двух слов на соответствие

```
import gensim,wget
from gensim.models import Word2Vec,KeyedVectors
import zipfile
model_url = 'http://vectors.nlpl.eu/repository/20/180.zip'
m = wget.download(model_url)
model_file = model_url.split('/')[-1]
with zipfile.ZipFile(model_file, 'r') as archive:
    stream = archive.open('model.bin')
    model = gensim.models.KeyedVectors.load_word2vec_format(stream, binary=True)
filename = 'model.bin'
model = KeyedVectors.load_word2vec_format(filename, binary=True)
words = ['день_NOUN', 'ночь_NOUN', 'человек_NOUN', 'семантика_NOUN', 'студент_NOUN', 'студент_ADJ']
print(model.similarity('да_NOUN', 'нет_NOUN')) 0.3658021
print(model.similarity('нет_NOUN', 'нет_NOUN')) 0.99999994
print(model.similarity('да_NOUN', 'конечно_NOUN')) 0.45536008
print(model.similarity('да_NOUN', 'хорошо_NOUN')) 0.3723208
print(model.similarity('да_NOUN', 'верно_NOUN')) 0.34742245
```

Добавление данных к анкете

Решение реализуется на языке Python3 при помощи Pandas

Сводная таблица по отфильтрованным данным из Postgresql, MySQL, Neo4j

	CRED_AMOUNT	CRED_TERM	CRED_OBJECT	CUST_MONTH_INCOME	CUST_FAMILY_MONTH_INCOME	CLIENT_ID	CUST_BIRTH	GUARANTOR_FLAG	GUARANTOR_BIRTH	PLEDGE_AMOUNT
0	100500.0	5	Выдача средств	35000.0	75000.0	100200300	1982	f		1970 0.0
1	200000.0	12	Кредит на обучение	0.0	0.0	100200301	1984	t		1984 0.0
2	55000.0	6	Выдача средств	0.0	0.0	100200302	1966	f		1970 0.0
3	15040000.0	25	Городская недвижимость	435000.0	0.0	100200303	2001	t		1999 21509000.0
4	654600.0	15	Оборудование	0.0	0.0	100200304	1968	t		1983 150000.0
5	2267000.0	20	Авто	120.0	157.0	100200305	1990	t		1984 3200000.0
6	559000.0	6	Земельный участок	66.0	150.0	100200306	1984	f		1970 680000.0
7	1547000.0	15	Авто	160.0	0.0	100200307	1949	t		1991 1599000.0
8	8838925.0	36	Дача	95.0	143.0	100200308	1984	t		1988 0.0
9	502000.0	20	Выдача средств	0.0	0.0	100200309	1988	f		1970 0.0
10	35500.0	5	Бытовая техника	0.0	0.0	100200310	2000	f		1970 0.0
11	987400.0	24	Авто	65.0	108.0	100200311	2002	t		1965 0.0
12	2502000.0	35	Коммерческая недвижимость	210.0	210.0	100200312	1999	t		1970 2830000.0

CUST_RELATION_BANK_TYPE	APP_SALE_CHANNEL	CITIZENSHIP	LOAN_ID	BEGIN_DT	CLOSE_PLAN_DT	CLOSE_FACT_DT	REPAYMENT_MODE	INTEREST_RATE	TURN_MONTH_AMT	SALARY_MONTH_INCOME	INSURANCE_CONTRACT_ID	CONTRACT_INSURANCE_AMT
	0	Онлайн-приложение	РФ	15556671000	2019	2029	0	аннуитет	12.5	0.0	0.0	800800100.0
	0	Онлайн-приложение		0	0	0	0	0	0.0	192394.0	145000.0	0.0
	0	0	Республика Беларусь	15556671001	2021	2023	0	дисконт	7.8	0.0	0.0	800800101.0
Инсайдер		Офис		0	0	0	0	0	0.0	77074.0	46750.0	0.0
	0	0	РФ	15556671002	2021	2022	0	аннуитет	11.3	0.0	0.0	800800104.0
	0	Сайт		0	0	0	0	0	0.0	0.0	0.0	0.0
	0	Онлайн-приложение	РФ	15556671003	2020	2022	2021	аннуитет	10.0	0.0	0.0	800800103.0
	0	Сайт		0	0	0	0	0	0.0	0.0	0.0	0.0
Сотрудник банка		0	РФ	15556671004	2016	2021	0	аннуитет	9.4	0.0	0.0	0.0
	0	Офис		0	0	0	0	0	0.0	0.0	0.0	0.0
	0	Онлайн-приложение	РФ	15556671005	2020	2021	2021	аннуитет	6.9	0.0	0.0	0.0
	0	Онлайн-приложение		0	0	0	0	0	0.0	35986.0	2500.0	0.0
	0	Сайт		0	0	0	0	0	0.0	0.0	0.0	0.0

Выявление параметров, значимых для кредитоспособности, из объединенного массива данных

Значимость параметров рассчитать на основе существующих данных о выплатах, к которым применены алгоритмы для ML

Идеальный пример: Python 3, использовать sklearn, взять Random Forest, GridSearchCV и посмотреть на feature_importances_

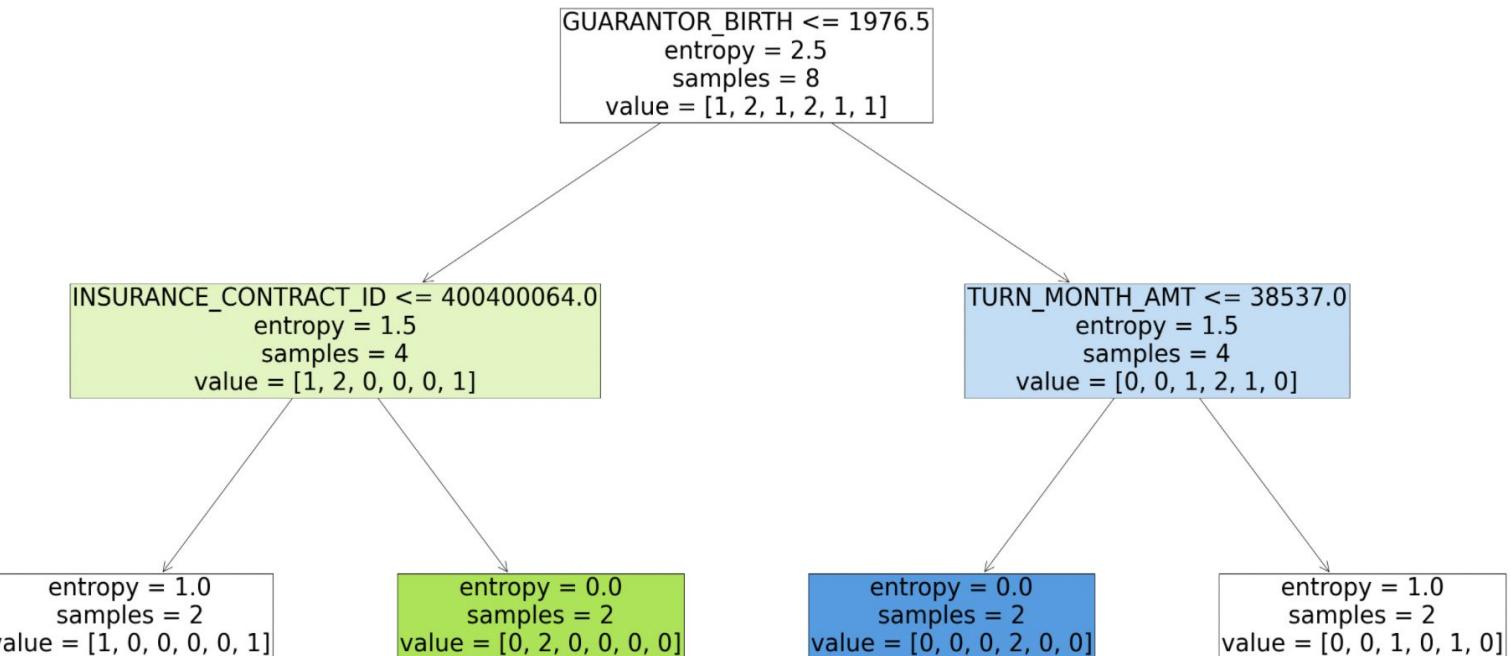
Алгоритм подготовки данных:

- 1) Перевести данные формата string/object в числовые (pd.get_dummies)
- 2) Выбрать колонку с данными, значение которых необходимо предсказать (нас интересуют данные о погашении вовремя, или о просрочках)
- 3) Применить подходящий алгоритм
- 4) Получить значимые параметры для системы скоринга

Пример применения машинного обучения:

Классифицировать клиентов срока кредита дополненной анкете методом Decision Tree (параметр “CRED_TERM”)

(данных из примера недостаточно для применения Random Forest)



Финальный продукт



Учитывает данные из разных источников



Значимые параметры для кредитоспособности клиентов



Список таких параметров может быть опубликован в открытом доступе и способствовать политике прозрачности

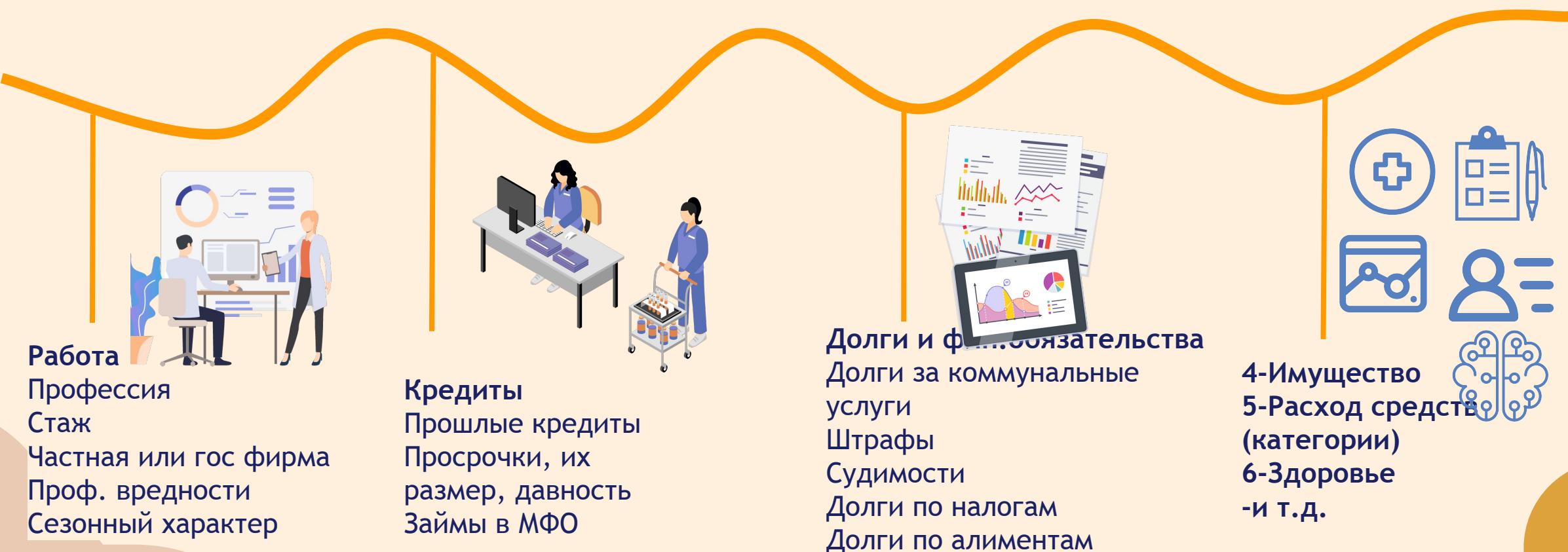


Как применить продукт к маркетинговым целям банка?

Может ли рассмотрение дополнительных данных улучшить продукт?

Roadmap: 1 дополнительные данные для повышения эффективного скоринга

Onay C., Öztürk E. A review of credit scoring research in the age of Big Data // Journal of Financial Regulation and Compliance. - 2018.



Roadmap: 2 добавление балльной системы после зелёного флага

Abellán J., Castellano J. G. A comparative study on base classifiers in ensemble methods for credit scoring /Expert systems with applications. - 2017. - Т. 73. - С. 1-10.

Наличие работы



Профессия
Стаж
Частная или гос.
фирма
Проф. вредности
Сезонный характер

-до 1 года +
-1-5 лет ++
-5 лет и более +++

-учитель, менеджер, оператор +++
-водитель++
-сапёр, пожарный +

Если на нашей выборке такое предположение подтверждается, опции, например: увеличить страховку

Внедрение балльной системы

```
# пример скрипта

s = 0 # кол-во баллов

credit = 1000000 # размер кредита в рублях

# данные о клиенте №1 ( Иванов Иван Иванович )

# персональные данные
data_1_pers = [
    'man', # пол
    40, # возраст
    True, # брак
    3 # кол-во детей
]

# работа и тд
data_1_fin = [
    100000, # официальная з.п.
    'manager_1', # должность и класс профессии
    7, # срок работы на последнем месте
    1, # кол-во офиц работ
    { 'car': 1200000,
      'house': 5000000 } # список имущества в собственности и его оценочной стоимости
]

# кредиты
data_1_credit = [
    0, # кол-во активных кредитов
    { 'car': 'данные о кредите',
      'house': 'данные о кредите' } # кол-во закрытых кредитов
]

# анализ соц сетей и др
data_1_live = [
    10, # результат оценки оплаты штрафов ( от 0 до 10)
    10, # результат анализа соц сети ( от 0 до 10 )
]
```

```
# обработка персональных данных
if 27 >= data_1_pers[1] >= 18:
    s += 10
elif 28 >= data_1_pers[1] >= 35: # оценка возраста
    s += 15
elif 35 >= data_1_pers[1] >= 55:
    s += 20
else:
    s += 13

if data_1_pers[2]:                      # социальное положение
    s += 20

if data_1_pers[0] == 0:
    s += 15
# обработка данных о работе

s += round((data_1_fin[0] / credit) * 150)

a = data_1_fin[1].split('_')[-1]
if a == '1':
    s += 150
if a == '2':
    s += 100
if a == '3':
    s += 50

if 3 >= data_1_fin[2] >= 1:
    s += 50
if 9 >= data_1_fin[2] >= 4:
    s += 100
```

```
if 3 >= data_1_fin[2] >= 1:
    s += 50
if 9 >= data_1_fin[2] >= 4:
    s += 100
else:
    s += 150

if 2 >= data_1_fin[3] >= 1:
    s += 100
else:
    s += 50

a = data_1_fin[4]
x = 0
for i in a:
    x += a[i] # СЛОУТ все имущества

if x >= credit:
    s += 50

if data_1_fin[5] == 0:
    s += 150 # кол-во активных кредитов

for i in range(len(data_1_live)):
    s += data_1_live[i]

print(s) # 618 баллов получает наш пользователь
```

Roadmap 3: изменение первичного скоринга литературных данных, на основе положительных и отрицательных заемщиков (страны/региона)

Albareto G., Felici R., Sette E. Does credit scoring improve the selection of borrowers and credit quality? //Bank of Italy Temi di Discussione (Working Paper) No. - 2016. - Т. 1090.



Профессия
Стаж
Частная или гос.
фирма
Проф. вредности
Сезонный характер

БЫЛО

-до 1 года +
-1-5 лет ++
-5 лет и более +++

-государственная ++
-частная+

-учитель, менеджер,
оператор +++
-водитель++
-сапёр, пожарный +

СТАЛО

-до 1 года +++
-1-5 лет ++
-5 лет и более +

-государственная +
-частная+++

-учитель, менеджер,
оператор +
-водитель+
-сапёр, пожарный +++

Маркетинговое предложение сейчас



Хоромы?!

Тут бы однушку
в
Подмосковье....



Маркетинг #1 - для различных категорий заемщиков

- Кластеризация заемщиков по выбранным параметрам (профессия, категории расходов, возраст и тд.)
- Разные предложения для неплательщиков и тех, кто платит вовремя

Маркетинг #2 гос. программы

<https://programs.gov.ru>



**ПОРТАЛ
ГОСПРОГРАММ РФ**

Для кого:

Молодые семьи

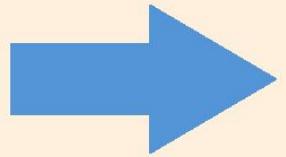
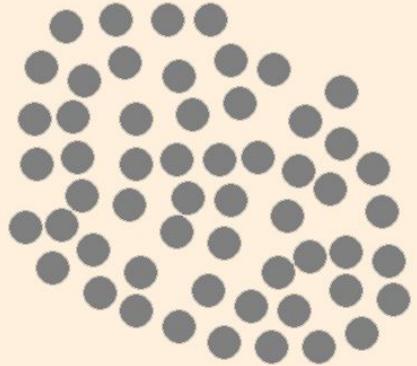
Многодетные родители
военных
Учёные

На что:

Новостройки

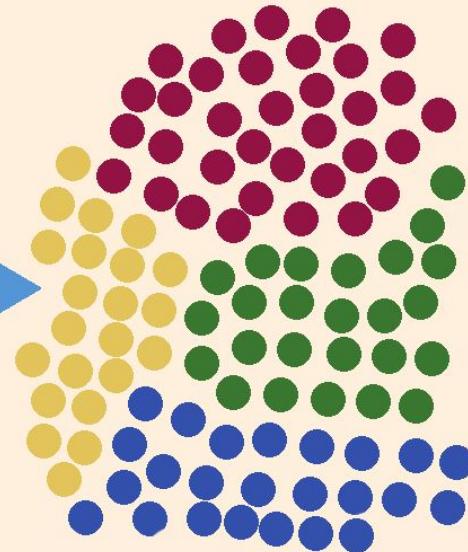
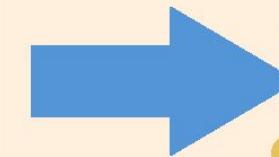
Сельское хозяйство
Гектар и дальние земли

Пример кластеризации заемщиков



Кластеризация
по возрасту

- от 18 до 27
- от 28 до 40
- от 41 до 50
- ≥ 51



Персонализированное маркетинговое предложение



Ой!! Беру!!

СМС: Уважаемый кот Василий! ПАО
Котобанк предлагает ипотеку на
хорошенькую однушку, мы можем
одобрить вам кредит без первоначального
взноса!



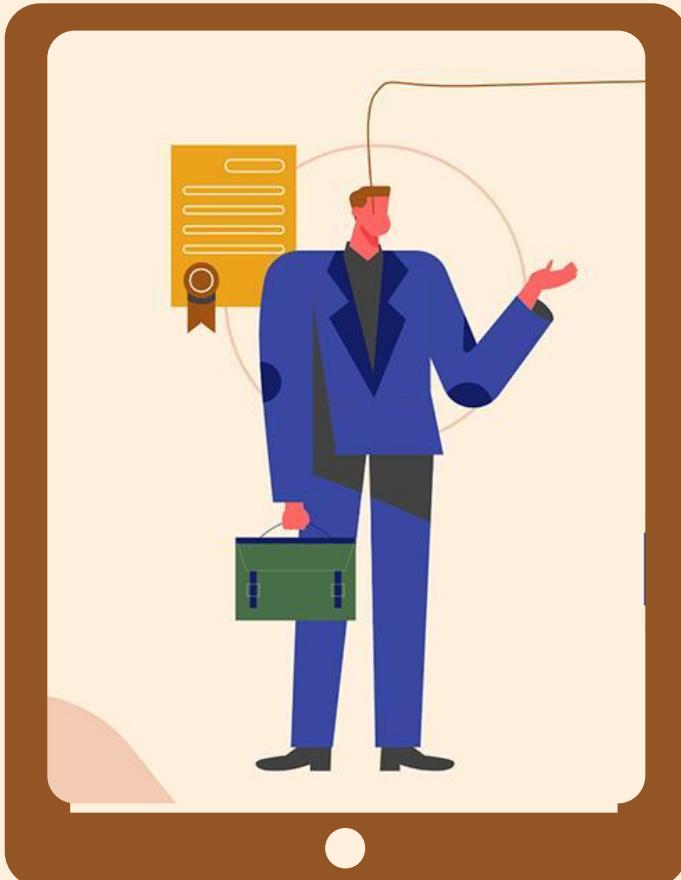
Решение представила команда Mental:



Прудий Дмитрий Игоревич



Красавин Георгий
Сергеевич



Контактные данные
89175881896
vasin-ks@rambler.ru



Зеленова Мария
Александровна



Васин Кирилл
Сергеевич

Спасибо за внимание!

Наобучают там
свои машины,
а кота кто
погладит

Усы, лапы и хвост -
моя
платежеспособность!

