

САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
ФАКУЛЬТЕТ ПРИКЛАДНОЙ МАТЕМАТИКИ – ПРОЦЕССОВ УПРАВЛЕНИЯ
КАФЕДРА ТЕХНОЛОГИИ ПРОГРАММИРОВАНИЯ

Чуриков Никита Сергеевич

Выпускная квалификационная работа бакалавра

**Предсказание атрибутов документов
в системе документооборота**

Направление 010400

Прикладная математика и информатика

Заведующий кафедрой,
кандидат технических наук,
доцент

Блеканов И. С.

Научный руководитель,
кандидат физ.-мат. наук,
доцент

Добрынин В. Ю.

Рецензент,
руководитель научной лаборатории Digital Design

Ашихмин И. А.

Санкт-Петербург

2017

Содержание

Введение	4
Обзор литературы	4
Постановка задачи	6
Глава 1. Обзор Docsvision	6
Глава 2. Загрузка данных	8
§2.1. Данные из document.json	9
§2.2. Данные из resolution.json	10
Глава 3. Описание алгоритмов машинного обучения используемых в работе	14
§3.1. Random forest	14
§3.1.1 Выделение важных атрибутов	14
§3.2. Stochastic gradient descent	15
§3.3. Линейная регрессия	16
§3.3. Word2vec	16
§3.4. LabelEncoding	17
§3.5. One Hot Encoding	17
§3.6. Алгоритмы понижения размерности	18
§3.6.1 PCA	18
§3.6.2 MDS	19
§3.6.3 TSNE	19
§3.7 Random oversampling	20
Глава 4. Метрики	20
Глава 5. Решения задач	22
§5.1. Подготовка категориальных переменных	22

§5.1.1 Понижение размерности	23
§5.2. Рекомендация исполнителя	25
§5.3. Прогнозирование времени исполнения документа	26
Глава 6. Эксперименты	26
§6.1. Прогнозирование времени исполнения документа	26
§6.2. Рекомендация исполнителя	29
§6.2.1 Первый уровень исполнения	31
§6.2.2 Второй уровень исполнения	34
§6.2.3 Третий уровень исполнения	39
Выводы	42
Заключение	42
Список литературы	42

Введение

Система электронного документооборота (СЭД) [1] – это система автоматизации процессов работы с документами, поддерживающая основные функции документооборота, такие как: хранение, создание, поиск.

Первые продукты такого рода появились в 80-х годах прошлого века, в виде запасного хранилища бумажных документов.

Позже, была добавлена функциональность для работы с электронными документами, такими как: электронная почта, поручения, документы, изображения и т.д.

И сегодня современные СЭД стараются идти в ногу со временем, добавляя такую функциональность, как извлечение текста из изображений [2] и автоматическая классификация текстовых документов [3].

В представленной дипломной работе предлагается два новых подхода к расширению функциональности СЭД Digital Design [4] Docsvision [5]:

1. Рекомендация исполнителя для задания;
2. Прогнозирование времени, которое потребуется на обработку документа.

Для решения этих задач компания Digital Design предоставила набор документов правительства Мурманской области за два года. На их основе был построен прототип системы обработки документов.

Обзор литературы

Для повышения теоретических знаний в области машинного обучения и информационного поиска, использовалась работа «Введение в информационный поиск» К. Маннинга [6], и в первую очередь глава 14 «Классификация в векторном пространстве» и глава 18 «Разложение матриц и латентно-семантическое индексирование». Для развития практических навыков в области машинного обучения и анализа данных использовалась книга В. Маккини

«Python для анализа данных» [7]. В ней последовательно проиллюстрированы подходы для манипулирования и визуализации данных в python. Богатая коллекция решений классических задач представлена на сайте kagge [8]. В частности, для построения модели регрессии, полезен разбор задачи по прогнозированию цен за дома [9]. Для ознакомления с алгоритмом Random forest, очень помогла классическая работа Breiman L. «Random forests» [10], а также для понимания того, как считается информативность атрибутов в алгоритме использовалась статья Louppe, G. «Understanding variable importances in forests of randomized trees» [11]. В этой статье приводится достаточно подробное описание того, как в случайных лесах выделяются важные атрибуты. В понимании задачи и структуры данных, в предоставленном датасете, значительную роль сыграла документация API Docsvision и интерактивная демонстрация создания документа в системе Docsvision [12]. О борьбе с несбалансированностью данных подробно и практично описано в статье «8 методов борьбы с несбалансированностью данных» [13]: в этой статье приводятся методики по уменьшению несбалансированности данных, такие как oversampling (генерирование атрибутов малых классов) и разбиение выборки на сбалансированные подвыборки.

Постановка задачи

В контексте данной работы, под документом будем понимать следующее:

Документ – это история обработки некоторого официального документа или поручения. В системе электронного документооборота Docvision он описывается двумя обязательными файлами содержащими метаданные: `document.json` и `resolution.json`, и одним или несколькими дополнительными файлами, которые могут быть изображением, текстовым файлом, презентацией и т.п.

Рассмотрим задачи, которые стоят перед нами:

1. С помощью методов машинного обучения необходимо построить модель, которая будет рекомендовать сотрудника-исполнителя задачи исходя из данных документа;
2. Построить модель, для прогнозирования длительности исполнения документа.

Специфика обеих задач в том, что для их решения в качестве атрибутов используются категориальные переменные. Поскольку стандартные методы, вроде One hot encoding подходят при небольшом количестве значений, необходимо найти иной способ представления переменных такого рода.

Глава 1. Обзор Docsvision

Перед тем, как перейти к данным, для наглядности, обсудим то, как выглядит и работает система документооборота компании Digital Design Docsvision. Схема работы с документом представлена на Рис. 1. При создании документа, есть возможность указать его тип (письмо, документ, и т.д.), краткое описание, от кого пришел файл, кому, автор документа, подразделение, в котором будет зарегистрирован документ и его категория. Это данные,

которые хранятся в файле **document.json**.

Созданное задание представлено на Рис. 2. В созданном задании, есть такие параметры как время начала и окончания, кто будет работать над заданием и описание поручения. Эти данные хранятся в файле **resolution.json**.

В следующей секции будет рассмотрена структура этих файлов, как они выглядят, какие переменные будут извлекаться.

DIGITAL DESIGN быстрый поиск ← Расширенный поиск

Регистрируется №вр-871
Вх. письмо №4223

[Регистрационные данные](#) [Резолюция](#)

⚠ Данные заполнены автоматически. Необходимо проверить

Письмо

Возможность кредитования предприятия по разработке Тюменского месторождения

От кого ФГУП СКТБ "Аметист" – Ковров И.П. ✕

Кому Арканянский И.В. ✕ Симонова С.А. ✕

Временный № вр-871 от 16.06.2016

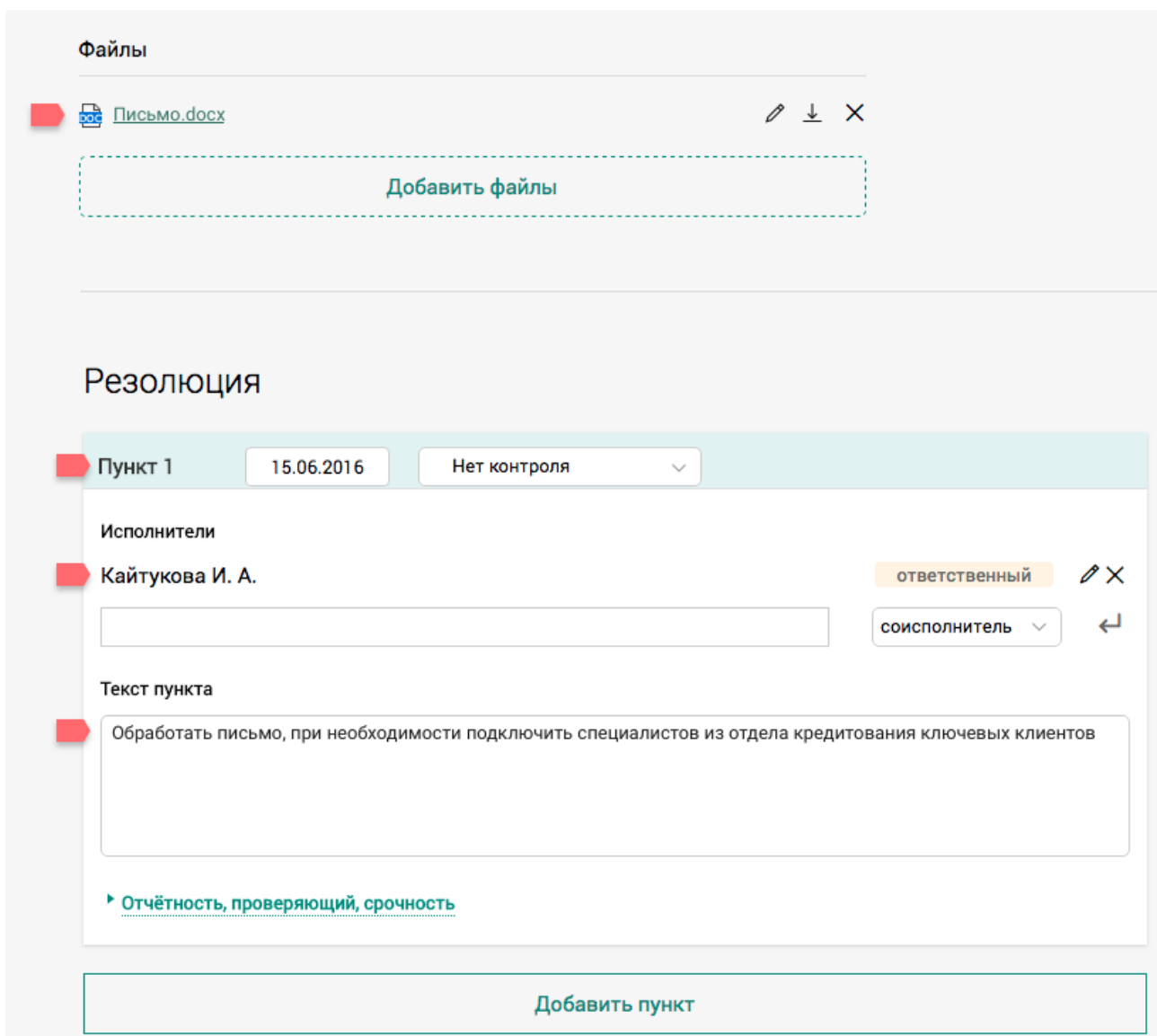
Автор Филиппенко И. Н.

Подразделение регистрации Направление СДУ

Категории Кредитование

Добавить дело

Рис. 1. Созданный документ



Файлы

Письмо.docx

Добавить файлы

Резолюция

Пункт 1 15.06.2016 Нет контроля

Исполнители

Кайтукова И. А. ответственный

соисполнитель

Текст пункта

Обработать письмо, при необходимости подключить специалистов из отдела кредитования ключевых клиентов

Отчётность, проверяющий, срочность

Добавить пункт

Рис. 2. Создание задания

Глава 2. Загрузка данных

Каждый документ в СЭД Docsvision описывается с помощью метаданных, хранящихся в файлах **document.json** и **resolution.json**. Эти файлы имеют фиксированную структуру и каждый объект в файле имеет свой фиксированный набор атрибутов. В основном, это категориальные переменные, но среди них имеются значения, уникальные для организации и значения фиксированные в системе.

Категориальные переменные задаются в виде UUID [14] – universally unique identifier, которое является уникальным значением, принимающим следующий вид: *xxxxxxxx-xxxx-Mxxx-Nxxx-xxxxxxxxxxxx*, где *M* и *N* задают версию идентификатора, а *x* является случайной латинской буквой или цифрой ($[a-z0-9]$). Вероятность повторения этих идентификаторов очень низкая и поэтому их используют для кодирования различных объектов в промышленном программировании. Т.е. у каждого объекта, вроде категории, человека, документа, файла и т.д. имеется свое уникальное значение.

Значения, фиксированные в системе, кодируются целочисленными значениями и едины для всех организаций.

§2.1. Данные из `document.json`

В файле `document.json` хранятся метаданные описывающие документ заданные при его регистрации в системе: категория, регистратор, краткое описание и т.д.

Представленные в файле `document.json` метаданные рассматриваются в данной работе как категориальные переменные.

Рассмотрим статистику переменных из **`document.json`** представленную в Таблице 1. Краткое описание каждой из переменных приведено в столбце **Описание**. Со структурой можно ознакомиться в Приложении А.

По данным Таблицы 1 видно, что у нас имеется восемь переменных, семь из которых – категориальные, одна переменная является строкой, а также *Categories* представляет из себя список.

Название	Тип	Количество	Описание
Content	string	7193 words	Краткое описание документа
Urgency	UUID	4	Устанавливает срочность документа: Обычная, Срочно, В.срочно, Оперативно
AccessType	UUID	3	Уровень доступа документа: Общий, Конфиденциально, ДСП
Categories	List of UUIDs	157	К какой категории относится документ
RegistrarDepartment	UUID	66	Департамент, в котором был зарегистрирован документ
Registrar	UUID	159	Тот, кто зарегистрировал документ
Operator	UUID	145	Тот, кто создал документ
Addressee.Reference	UUID	5007	Ссылка на человека или отдел

Таблица 1. Статистика переменных

§2.2. Данные из resolution.json

Файл resolution.json содержит в себе сведения о прохождении документом цепочки заданий на исполнение в организации.

Задания в файле **resolution.json** имеют рекурсивную структуру, пример которой представлен в листинге 1.

```

"Tasks": [{
  "Parts": [{
    "Executors": [
      "Tasks": [...],
    ]
  }],
  "Executes": {,
    "Name": "sample string 12",
    "Id": "200e825a-796b-4580-bb3d-417e7676d2fa"
  },
  "Appointed": {
    "Name": "sample string 12",
    "Id": "200e825a-796b-4580-bb3d-417e7676d2fa"
  },
},
]

```

Listing 1. Рекурсивная структура заданий

Имеется два вида исполнителя: *Appointed* и *Executes*. Первый обозначает человека, на которого было назначено задание, а второй – непосредственного исполнителя задания. Исполнители могут отличаться в том случае, если

начальник, на которого назначено задание, передал его своему подчиненному для непосредственного исполнения.

Рассмотрим распределение количества выполненных заданий для этих двух переменных на Рис. 3.

Видно, что в данном датасете всего в 43 случаях задание выполнил не тот, кто был назначен. В семи случаях это обращение граждан, в тридцати – письмо и пяти – поручение. Эти результаты подтверждают, что обычно, если человек, которому назначено задание и тот кто непосредственно выполнил, отличаются, то в подавляющем большинстве случаев это начальник передал задание подчиненному.

Не трудно заметить, что распределения очень похожи, но это не значит, что эти значения всегда одинаковые, потому что из 370000 заданий, в 43 случаях *Appointed* и *Executes* разные.

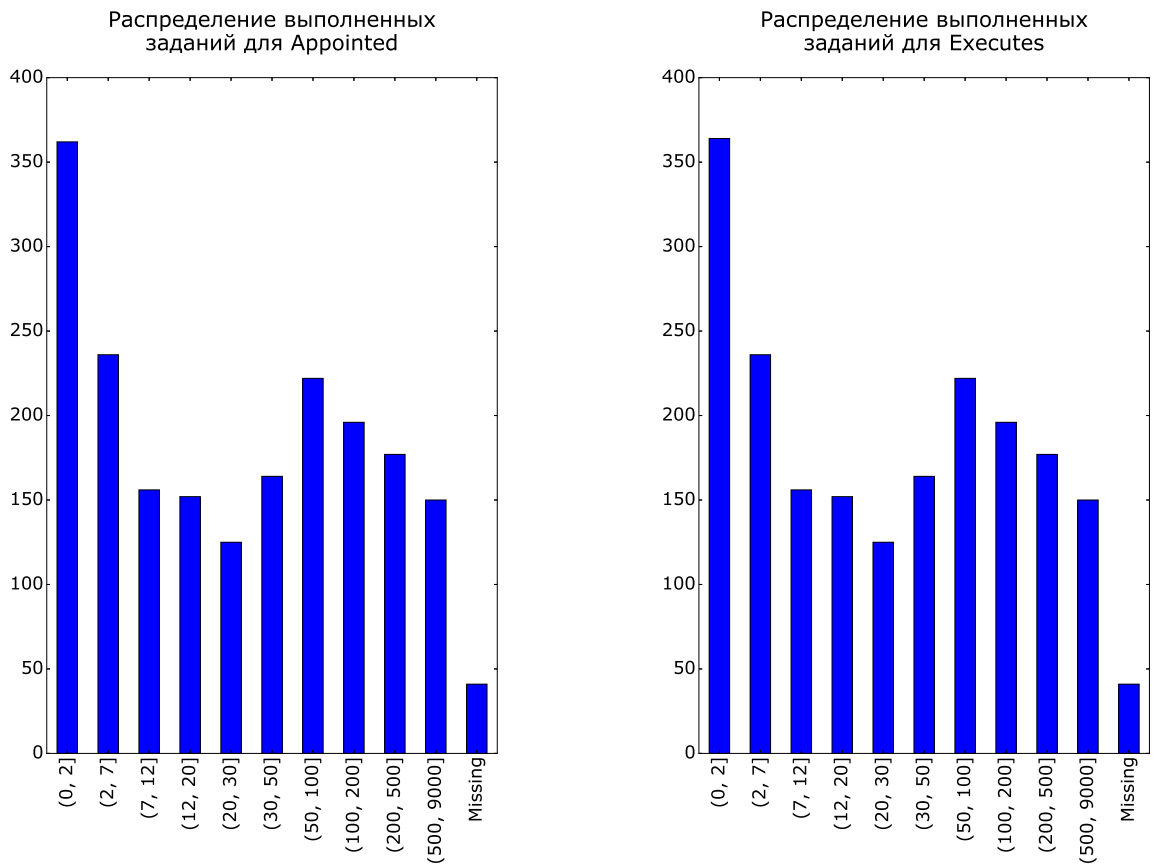


Рис. 3. По оси X интервалы с количеством выполненных заданий, а по Y сколько заданий в каждой корзине

Вспомним теперь, что задания имеют рекурсивный характер и могут иметь много уровней вложенности. Посмотрим, сколько всего может быть уровней назначения:

Уровень	Количество заданий на уровне
1	115772
2	154495
3	69980
4	24802
5	5365
6	706
7	51
8	8

Таблица 2. Количество уровней исполнения в резолюции

Как можно увидеть в таблице 2 всего имеется 8 уровней в представленном датасете. На Рис. 29, в Приложении С, представлено, сколько заданий выполнено для соответствующих уровней.

Нами будут рассмотрены только первые 3 уровня, представленные на Рис. 4, 5, 6.

На графиках, интервалы по оси y определяют количество заданий выполненных исполнителями, а по оси x – количество исполнителей, попавших в этот интервал.

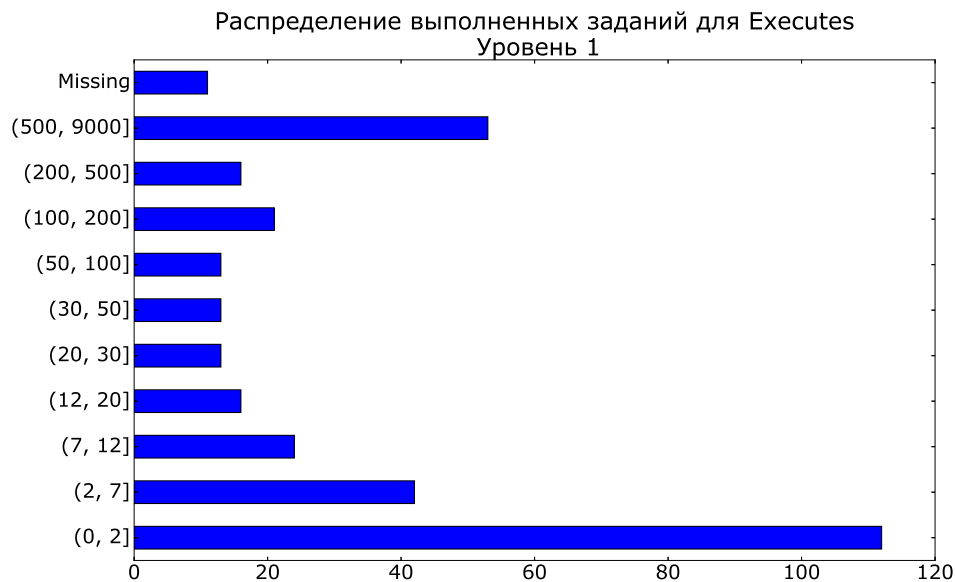


Рис. 4.

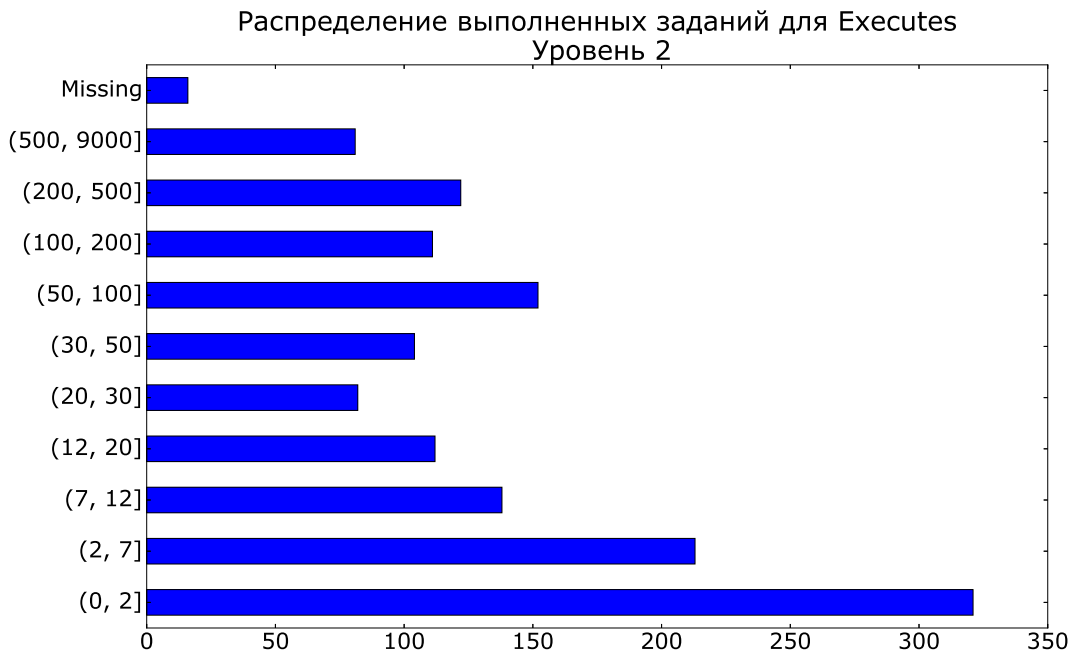


Рис. 5.

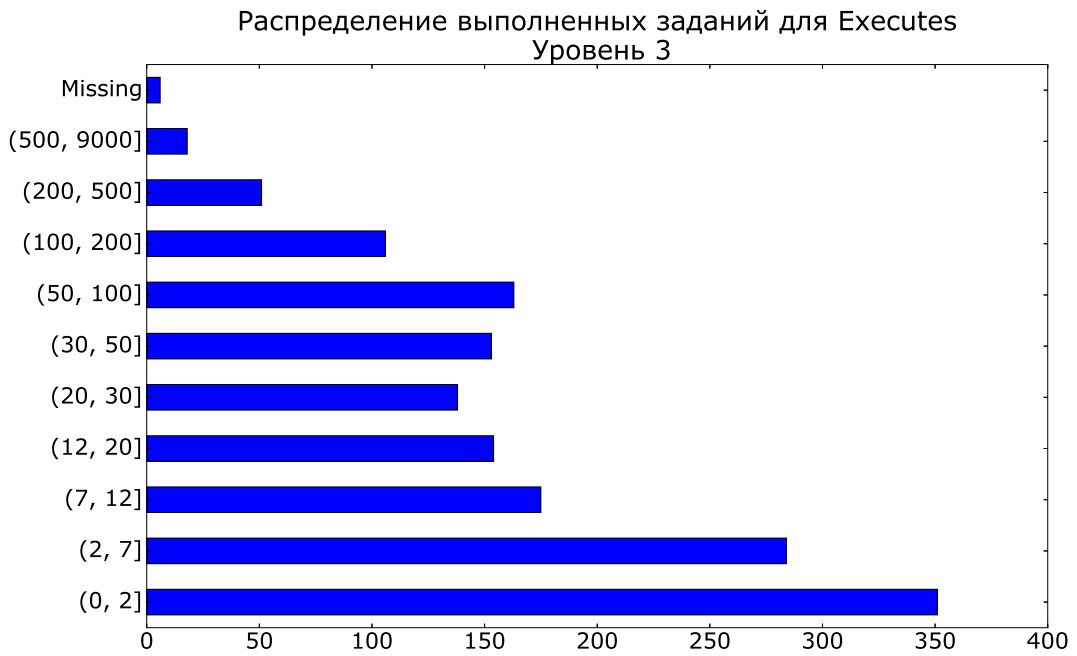


Рис. 6.

Глава 3. Описание алгоритмов машинного обучения используемых в работе

§3.1. Random forest

Алгоритм Random forest [15, 16, 10] является сегодня универсальным средством для решения задач связанных с машинным обучением. Основанный на ансамбле решающих деревьев, его можно использовать как для задач классификации, так и для регрессии. Он довольно прост для понимания, устойчив к переобучению и реализован на многих языках программирования.

Однако, несмотря на то, что в теории, он способен работать с любыми категориальными переменными, на практике, если передавать ему классы закодированные через Label Encoder, то он будет считать, что классы 1 и 2 близки, а 9 и 10 далеки от них. Поэтому, как правило, используют One hot encoding.

Данный метод был выбран, потому что он не зависит от масштаба данных.

Воспользуемся реализацией данного алгоритма для регрессии и классификации из библиотеки sklearn [17]. Из параметров, для классификации, установим `class_weight='balanced'`, остальные параметры оставим по умолчанию.

§3.1.1. Выделение важных атрибутов

В моделях, основанных на решающих деревьях, можно выделить атрибуты, которые приносят наибольшее количество информации [11].

$$Imp(X_m) = 1/N_T \sum_T \sum_{t \in T: v(s_t)=X_m} p(t) \Delta i(s_t, t)$$

где N_T – количество узлов в дереве T , t – узел в дереве, $p(t) = \frac{N_t}{N}$ – количество переменных, достигающих узел t , $v(s_t)$ – переменная, определяющая разби-

ение в узле t . $\Delta i(s_t, t)$ определяет величину изменения неопределенности в узле t :

$$\Delta i(s_t, t) = i(t) - p_L i(t_L) - p_R i(t_R)$$

где $i(t)$ определяет коэффициент неопределенности (коэффициент Gini, энтропию или дисперсию целевой переменной Y), $p_L = \frac{N_{tL}}{N_t}$ и $p_R = \frac{N_{tR}}{N_t}$ определяют пропорции: сколько переменных идут в левый или правый узел соответственно.

В качестве коэффициента неопределенности $i(t)$ был выбран коэффициент Gini.

$$i(t) = 1 - \sum_{c \in C} p_c^2, \text{ где } p_c - \text{вероятность класса } c.$$

Реализация этой меры есть в классе `RandomForest`, в библиотеке `sklearn`.

§3.2. Stochastic gradient descent

Стохастический градиентный спуск [18, 19] простой и эффективный подход к обучению линейных классификаторов.

Математическая формулировка: имея набор тренировочных данных:

$$\{(x_i, y_i)\}_{i=1}^N, \text{ где } x_i \in \mathbb{R}^n \text{ и } y_i \in \{0, 1\}$$

необходимо построить линейную функцию $f(x) = w^T x + b$ с весами $w \in \mathbb{R}^m$ и порогом $b \in \mathbb{R}$. Типичный способ нахождения весов – минимизация целевой функции при помощи градиентного спуска на тренировочном множестве.

$$E(w, b) = \frac{1}{n} \sum_{i=1}^n L(y_i, f(x_i)) + \alpha R(w)$$

где L – это функция потерь, которая обучает модель, а R – это параметр регуляризации (поправка), которая штрафует модель, $\alpha > 0$ – неотрицательный гиперпараметр.

Разные L соответствуют разным классификаторам. Мы будем брать L , которая соответствует логарифму от функции правдоподобия

$$L = \sum_{i=1}^n \log \left(\frac{1}{\exp(-y_i f(x_i)) + 1} \right)$$

Данный метод хорошо подходит для данных, которые находятся в одном масштабе, поэтому будем использовать его для значений \mathbf{X} , полученных с помощью One hot encoding.

Реализацию этого метода возьмем из библиотеки `sklearn`. Из параметров, установим `class_weight='balanced'`, остальные параметры оставим по умолчанию.

§3.3. Линейная регрессия

Линейная регрессия [20] – это классический алгоритм для прогнозирования данных. Ее задача – построить прямую (гиперплоскость в случае многих переменных) через точки из тренировочного множества по заданным наблюдениям.

Реализация данного алгоритма была взята из библиотеки `sklearn` со стандартными параметрами.

§3.3. Word2vec

Word2Vec [21, 22, 23] – технология от Google, которая нацелена на статистическую обработку больших массивов текстовой информации. W2V собирает статистику совместного появления слов в предложениях, а затем обучает нейронную сеть по алгоритму skip-gram, описанному в оригинальной статье. W2V предлагает подход к представлению слов, который учитывает их положение в предложении, в отличие от классического подхода – мешка слов. В результате получаются вектора слов в N -мерном пространстве.

Мы попробуем обучить модель на имеющихся текстовых данных (*Content*) и на выгруженных документах с сайта правительства [24, 25].

Реализация этого метода была взята из библиотеки `gensim` со стандартными параметрами.

§3.4. LabelEncoding

Данный подход интерпретирует категориальные переменные, как упорядоченный целочисленный список. Как правило, это не так и кодируя таким образом переменные, привносится лишняя информация.

Данный подход имеет серьезные недостатки такие как:

1. В данные привносится не существующая информация;
2. Нет возможности работать со списочными категориальными переменными.

Однако у метода есть серьезный плюс по причине которого он все таки рассматривается в данной работе: данный метод хранит данные относящиеся к одной переменной в одном столбце.

Метод проиллюстрирован в листинге 2.

```
In [17]: cats = ['Москва', 'Санкт-Петербург', 'Нижний-Новгород', 'Москва']
In [18]: from sklearn.preprocessing import LabelEncoder
In [19]: le = LabelEncoder()
In [20]: le_cats = le.fit_transform(cats)
In [21]: le_cats
Out [21]: array([0, 2, 1, 0])
```

Listing 2. Использование `LabelEncoder` из библиотеки `sklearn`

§3.5. One Hot Encoding

One Hot Encoding преобразовывает переменные в бинарные (или так называемые, `dummy values`). Т.е. каждое значение категориальных переменных будет иметь свой столбец, где каждая строка обозначает наличие значения переменной в документе (значение 1) или отсутствие (значение 0). Метод проиллюстрирован в листинге 3.

Данный метод также обладает недостатками:

1. в ходе его работы теряется связь между переменными;
2. при большом количестве значений значительно повышается размерность.

```
In [22]: cats = ['Москва', 'Санкт-Петербург', 'Нижний-Новгород', 'Москва']
In [23]: import pandas as pd
In [24]: pd.get_dummies(cats)
Out [24]:
```

	'Москва'	'Нижний-Новгород'	'Санкт-Петербург'
0	1.0	0.0	0.0
1	0.0	0.0	1.0
2	0.0	1.0	0.0
3	1.0	0.0	0.0

Listing 3. One Hot Encoding используя pandas

Однако указанные минусы в нашем случае можно компенсировать: в разделе ниже будут рассмотрены варианты, как измерить дистанцию между людьми и другими категориальными переменными.

§3.6. Алгоритмы понижения размерности

Рассмотрим методы, которые позволят нам сохранить информацию о категориальных переменных и при этом получить одномерное представление. Для работы этих методов, необходимо ввести меру расстояния между значениями. Нам необходимо построить матрицу расстояний для переменных и понизить ее размерность описанными ниже методами.

Так, например, с городами, в качестве меры расстояния можно было бы взять дистанцию двух городов или среднее время на дорогу, в зависимости от задачи.

§3.6.1. PCA

Метод главных компонент (PCA) [26, 27] является линейным методом и его основная идея, в том, чтобы спроецировать данные на гиперплоскость с наименьшей ошибкой проектирования или, иначе с сохранением большей части дисперсии данных.

Данный метод находить только линейные подпространства исходного пространства, которые «объясняют» данные с высокой точностью. Но, как правило, данные имеют нелинейную структуру, поэтому также будут использованы нелинейные методы понижения размерности.

Имплементация этого метода была взята из библиотеки `sklearn`, с установленным параметром количества компонент `n_components=1`, остальные параметры по умолчанию.

§3.6.2. MDS

Многомерное масштабирование (MDS) проецирует точки в пространство малой размерности, так чтобы минимизировать несходство между попарными расстояниями в исходном пространстве и пространстве малой размерности.

Данный метод был выбран, потому что он сохраняет *глобальную* структуру данных.

Реализация данного метода была взята из библиотеки `sklearn`, с установленным параметром количества компонент `n_components=1` и не сходством (`dissimilarity`), для структур, которые представляют из себя матрицу дистанций равным `'precomputed'`, а для матриц, где матрица дистанций не была построена `'euclidean'`.

§3.6.3. TSNE

Основная идея t-SNE [28] конвертировать близость каждой пары точек в исходном пространстве \mathbb{R}^D большой размерности в вероятность того, что одна точка данных связана с другой точкой так, как с ее соседом. Т в названии алгоритма означает, что при вычислении градиента используется t-распределение Стьюдента.

Было принято решение использовать этот метод, потому что он сохраняет *локальную* структуру данных.

Реализация данного метода была взята из библиотеки `sklearn`, с установленным параметром количества компонент `n_components=1` и мерой похожести `metric`, для структур, которые представляют из себя матрицу дистанций равным `'precomputed'`, а для матриц, где матрица дистанций не была построена `'euclidean'`, также, предварительно, вычисляется понижение размерности с помощью метода главных компонент (`init='pca'`).

§3.7. Random oversampling

Random oversampling применяется для задач классификации с несбалансированными классами. Его назначение в том, чтобы повысить количество наблюдаемых значений для редких классов, т.е. редкие наблюдения классов он случайным образом повторяет.

Однако, применение данного метода может вести к переобучению, если классы слишком несбалансированы.

Реализация данного метода была взята из библиотеки `imblearn`.

Глава 4. Метрики

Рассмотрим метрики для определения точности классификаторов и регрессии.

Для классификации будем использовать точность (P), полноту (R) и F_1 метрики:

$$P = \frac{tp}{tp + fp} \quad (1)$$

$$R = \frac{tp}{tp + fn} \quad (2)$$

$$F1 = \frac{2PR}{P + R} \quad (3)$$

Где (1) определяет долю релевантных документов среди найденных, (2)

– доля релевантных документов среди всех релевантных, (3) – среднее гармоническое между двумя величинами.

Эти величины получаются для бинарной классификации. При многоклассовой, для каждого класса будет посчитана своя точность, полнота и F_1 меры, а затем будет взято среднее значение для каждой из метрик, т.е.:

$$P_{multiclass} = \frac{\sum_{i \in C} P_i}{N_C} \quad (4)$$

Где C – классы в представленной выборке, N_C – количество классов.

Так как нам необходимо рекомендовать людей, то посмотрим, в скольких случаях релевантный исполнитель встречается в списке первых n исполнителей. Выбранные нами классификаторы (Random forest и логистическая регрессия с градиентным спуском) дают в качестве результата прогнозирования вектор вероятностей по всем классам. Поэтому можно отсортировать этот список и брать первые n интересующих значений. Обозначим отсортированный список классов по вероятностям как C^{sorted} , релевантный класс как c_r , матрицу вероятностей Y_{pred} размерности $[N_{test} \times N_C]$, где N_{test} – количество примеров в тестовой выборке, $C_{1:n}$ обозначим с первого по n -ый элементы из вектора C .

$$score_n(C^{sorted}) = \begin{cases} 1, & \text{если } c_r \in C_{1:n}^{sorted} \\ 0, & \text{иначе} \end{cases} \quad (5)$$

$$S_n = \frac{\sum_{i \in N_{test}} score_n(sort_{desc}(Y_i^{pred}))}{N_{test}}$$

Для задачи регрессии воспользуемся коэффициентом детерминации (6) и среднеквадратическим отклонением (7).

Пусть \hat{y}_i – спрогнозированные значения i -го наблюдения, y_i обозначает правильное значение, тогда R^2 определяется следующим образом для $n_{samples}$.

$$R^2(y, \hat{y}) = 1 - \frac{\sum_{i=1}^{n_{samples}-1} (y_i - \hat{y}_i)^2}{\sum_{i=1}^{n_{samples}-1} (y_i - \bar{y})^2} \quad (6)$$

$$\text{где } \bar{y} = \frac{1}{n_{samples}} \sum_{i=1}^{n_{samples}-1} y_i.$$

Коэффициент R^2 определяет, насколько хорошо модель прогнозирует примеры, которых не было в тренировочной выборке. Наилучшее значение коэффициента 1. В случае 0-го значения, модель всегда дает один и тот же результат, независимо от входных данных.

$$MSE(y, \hat{y}) = \frac{1}{n_{samples}} \sum_{i=1}^{n_{samples}-1} (y_i - \hat{y}_i)^2 \quad (7)$$

(7) Определяет ошибку регрессии. Чем меньше эта величина, тем выше точность прогнозирования.

Глава 5. Решения задач

Перед нами стоят серьезные задачи по подготовке данных и построению рекомендательной системы. Еще раз обсудим их:

1. Необходимо найти способ представить категориальные переменные в числовом формате, при этом постараться не потерять информацию, связанную с ними;
2. Научиться рекомендовать исполнителей по категориальным переменным;
3. Прогнозировать количество дней, которое потребуется на задание.

§5.1. Подготовка категориальных переменных

Как видно из Таблицы 1 данные, в основном, представляют из себя категориальные переменные и нам необходимо выбрать метод представления их в виде чисел. В главе 3 рассмотрены два стандартных подхода: Label Encoding и One Hot Encoding.

Мы предлагаем представлять категориальные переменные в виде векторов через понижение размерности матрицы дистанции между переменными.

Обычно методы понижения размерности используют для того, чтобы графически представить данные высокой размерности в трехмерном или дву-

мерном пространстве. Мы же собираемся отобразить данные на одномерное пространство, чтобы получить возможность передать данные методам машинного обучения, при этом не теряя и не внося новой информации.

Для таких переменных, как тип доступа (*AccessType*) или срочность (*Urgency*) можно применить Label Encoder, потому что для данных этого типа существует порядок подходящий для использования LabelEncoding, что видно из Таблицы 1.

§5.1.1. Понижение размерности

В наших данных значительную роль играют люди, поэтому логично найти какой-то способ измерить расстояние между ними. Построим матрицу связей \mathbf{D} размерности $[k \times k]$, где k – количество людей(отделов):

$$D_{i,j} = \begin{cases} 1/d_{i,j}, & \text{если } d_{i,j} \neq 0 \\ 1, & \text{иначе} \end{cases}$$

где значение $d_{i,j}$ равно количеству раз, i -ый сотрудник(отдел) встречается вместе с j -ым сотрудником(отделом). Например, если два человека работают в одном отделе и работали над одним документом, тогда $d_{i,j} = 2$ для этих двух людей.

Получается, чем больше два человека работали вместе, тем ближе значение $D_{i,j}$ к 0, а соответственно, если два человека работали вместе менее двух раз, то $D_{i,j} = 1$.

Такую матрицу \mathbf{D} можно построить для следующих переменных: *Operator*, *Registrar*, *RegistrarDepartment*, *Addressee*.

С переменными, у которых нельзя построить матрицу расстояний описанным выше способом(*Categories* и *Kind*) поступим следующим образом:

- Измерим расстояние между категориями через людей, работавших над категориями.

Построим матрицу категорий - людей \mathbf{M} , размерности $[n \times k]$, где n –

количество категорий, а k – количество людей, $m_{i,j}$ обозначает, сколько раз i -я категория, встретила с j -ым человеком.

Вычисляем каждый элемент матрицы по следующему правилу:

$$M_{i,j} = \begin{cases} 1/m_{i,j}, & \text{если } m_{i,j} \neq 0 \\ 1, & \text{иначе} \end{cases}$$

- Построим Word2Vec модель по кратким описаниям документов (*Content*) или по какому-нибудь датасету (в данном случае использовались, государственные документы с сайта правительства РФ [24]. Также в открытом доступе есть документы на лето 2016 года и обученная W2V модель [25]). В результате для каждого слова получим k -мерное представление в виде вектора. Обозначим обученную Word2Vec модель как $\mathbf{w2v}$, некоторое слово как w , словарь известных слов V . Построим матрицу \mathbf{M} следующим образом:

```

M ← матрица нулей[n × k];
for c ∈ Categories do
    sentences ← выбрать Content документов, с категорией c;
    s ← нулевой вектор размерности k;
    for sentence ∈ sentences do
        for w ∈ sentence do
            if w ∈ V then
                s ← s + w2v[w];
            end
        end
    end
    M.c ← s;
end

```

Algorithm 1: Алгоритм вычисления матрицы \mathbf{M} для *Categories* и *Kind*

Затем строим матрицу \mathbf{D} для *Categories* и *Kind* следующим образом:

$$D_{i,j} = \sqrt{\sum_{l=0}^{k-1} (M_{i,l} - M_{j,l})^2}$$

Имея матрицу расстояний \mathbf{D} можно понизить ее размерность и сопоставить каждой переменной число. Сделав это, получаем для каждой категориальной переменной некоторое число. Это значение, в дальнейшем, можно использовать как значения категориальных переменных в матрице \mathbf{X} .

§5.2. Рекомендация исполнителя

Дана матрица \mathbf{X} размера $[N \times K]$, где N – это количество заданий, а K – количество переменных, полученных либо понижением размерности, либо с помощью One hot encoding и вектор \mathbf{Y} размерности N .

Если обратиться к Рис. 4, 5, 6 то видно, что представленные классы несбалансированы. Есть сотни людей, кто был назначен что-то выполнять один раз, а есть те, у кого в истории по тысяче выполненных заданий. Поэтому применение алгоритмов классификации без предварительной борьбы с несбалансированностью – наивно и некорректно.

Во-первых разобьем весь датасет на уровни от 1-го до 3-го, во-вторых, на каждом уровне разобьем данные таким образом, чтобы частоты выполненных заданий исполнителями были как можно более сбалансированными, в-третьих, будем применим oversampling для исполнителей с малым количеством заданий. В-четвертых, при классификации будем используем флаги для весов классов `Classification(class_wight='balanced')`. Установка этого флага вносит следующий коэффициент для весов классов:

$$\frac{n_{features}}{n_{classes} * np.bincount(Y)}$$

Также исключим из рассмотрения атрибут *Addressee*, поскольку в данном датасете, более чем в 70% случаев он совпадает с *Appointed* – целевой переменной.

Для решения задачи рекомендации исполнителя, воспользуемся алгоритмами Random forest и стохастическим градиентным спуском с логистической функцией потерь.

§5.3. Прогнозирование времени исполнения документа

В системах, связанных с документами, имеются определенные сроки на обработку документа. Например, обращение гражданина должно рассматриваться не дольше тридцати дней. Поэтому важно выяснить, что, что больше всего влияет на длительность обработки документа, и возможно ли построить модель, которая определяла бы это время.

В файлах *resolution.json* у каждого задания есть следующие сроки: *CreationDateTime*: – время создания, *ActualTerm*: – время завершения задания. Для того, чтобы получить время исполнения документа, посчитаем среднее время затраченное на каждое из заданий.

В результате, для каждого документа, получим величину, соответствующую времени, которое потребуется на обработку документа.

Эти значения будем считать наблюдаемыми Y . А X возьмем как прежде, но добавим *Addressee* в атрибуты.

Для решения задачи прогнозирования времени выполнения документа воспользуемся алгоритмами Random forest и линейной регрессией.

Глава 6. Эксперименты

§6.1. Прогнозирование времени исполнения документа

Рассмотрим задачу прогнозирования длительности обработки документа.

Пусть X – матрица с атрибутами, полученная, либо с помощью One hot encoding, либо с помощью понижения размерности, а Y – вектор с со средним временем выполнения заданий в каждом документе.

На Рис. 7 представлено распределение количества дней, затраченных на обработку каждого документа.

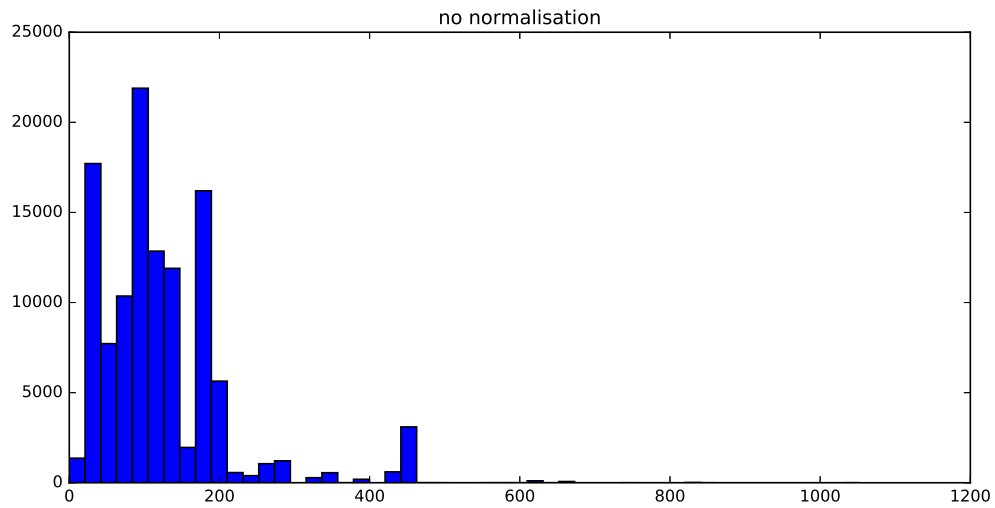


Рис. 7. Распределение количества дней

Регрессия дает лучше результаты, когда целевая переменная имеет малую дисперсию.

Стандартный метод для нормализации – функция $y' = \log(y + 1)$. Применяя ее, получается получается распределение, представленное на Рис. 8.

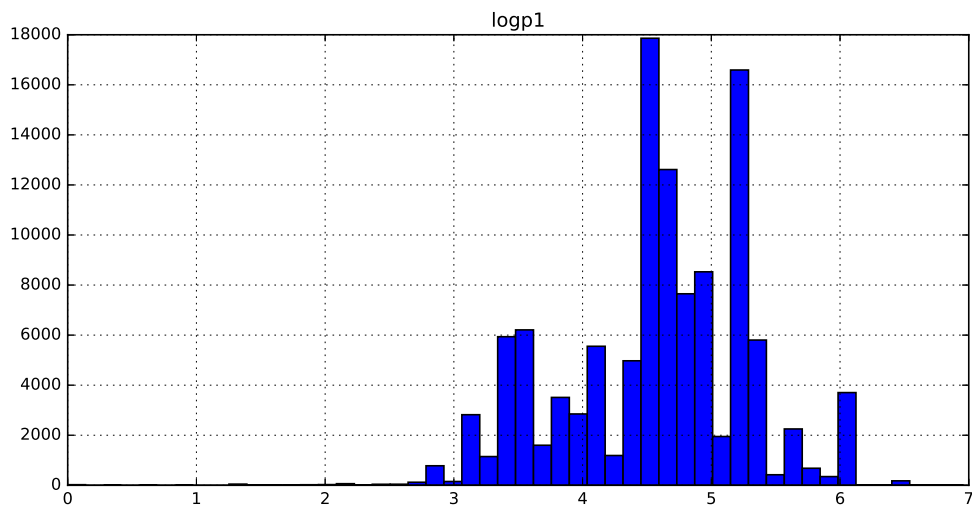


Рис. 8. Нормализованное распределение количества дней

Построим регрессию по атрибутам, полученным с помощью One hot encoding и понижения размерности. Матрица X_{dim_red} с атрибутами полученными в результате понижения размерности будет иметь размер $[115730 \times 23]$,

а матрица X_{ohe} полученная в результате One hot encoding, размерность $[115730 \times 357]$. Воспользуемся следующими категориальными переменными: *Operator*, *Registrar*, *RegistrarDepartment*, *Urgency*, *AccessType*, *Kind*.

В матрице X_{dim_red} воспользуемся всеми повышениями размерностей: (MDS, PCA, TSNE), а *Urgency*, *AccessType* преобразуем в численный вид с помощью Label encoding.

Для обучения модели возьмем 75% от всего набора данных. Результаты прогнозирования на тестовой выборке размера 25% от всего набора данных, представлены на Рис. 9.

Метрики были вычисленные на данных, преобразованных в исходный вид с помощью формулы $y = \exp(y') - 1$. На графике с результатами 9 введены следующие обозначения:

- RFR – random forest regressor;
- LinearRegression – линейная регрессия;
- `_ohe` – Использованы данные, полученные с помощью one hot encoding;
- `_dim_red` – Использованы данные, полученные с помощью понижения размерности.
- R^2 – мера R2 (6), чем больше, тем лучше.
- `error` – среднеквадратическое отклонение (7): чем меньше, тем лучше.

В результате видим, что данные, полученные с помощью понижения размерности, можно применять не только для нелинейных моделей, но также к линейным. Атрибуты, полученные с помощью one hot encoding при использовании в линейной регрессии дают $R^2 = -3.23338968469e + 20$ и среднеквадратическое отклонение равное 58488029338.625488. Это показывает, что нельзя применять линейную модель для категориальных переменных, полученных с помощью One hot encoding.

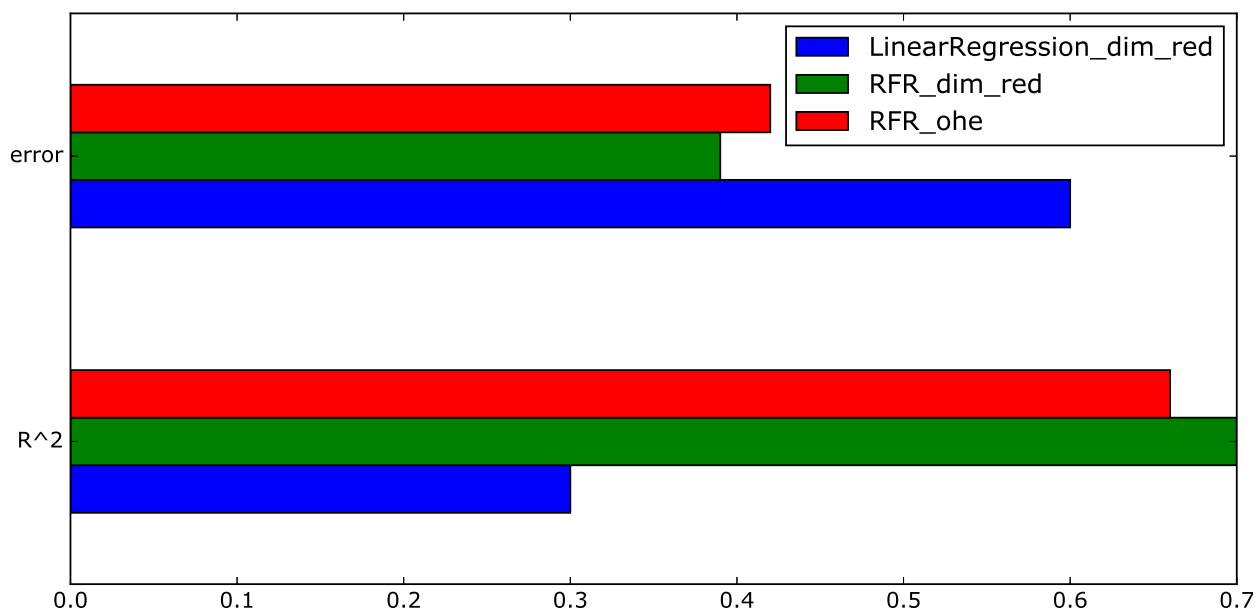


Рис. 9. Результаты применения регрессий.

§6.2. Рекомендация исполнителя

Решать задачу рекомендации исполнителя будем следующим образом: пусть X – матрица с атрибутами, полученная, либо с помощью one hot encoding, либо с помощью понижения размерности, а Y – вектор с исполнителями, которые являются категориями в данной задаче. Предварительно, если у документа было несколько исполнителей, то продублируем строки из матрицы X столько раз, сколько исполнителей для соответствующего документа.

Затем, выберем из этой матрицы первые три уровня исполнения (Рис. 4, 5, 6). Для второго уровня добавим в атрибуты исполнителей, которые были на первом уровне, а для третьего, соответственно, с первого и второго уровней.

Алгоритмы классификации будем обучать, как на данных полученных с помощью one hot encoding, так и с переменными, полученными в результате понижения размерности. Причем, учитывая тот факт, что этот метод способен выделять важные атрибуты, будем передавать ему все переменные полученные в результате понижения размерности (MDS, PCA, TSNE). В таблицах с результатами введем следующие обозначения:

- **subset** – выбранные исполнители, с количеством заданий из промежутка;
- **subset_train_size** – размер тренировочной выборки матрицы наблюдаемых значений для данных исполнителей;
- **subset_test_size** – размер тестовой выборки;
- **n_classes** – количество исполнителей, попавших в подвыборку **subset**;
- **method** – выбранный метод классификации и понижения размерности, где
 - RFC – random forest classifier;
 - SGD – stochastic gradient descent;
 - **_ohe** – one hot encoding;
 - **_dim_red** – понижение размерности;
 - **w** или **w/o oversampling** – с или без повторения наблюдаемых значений.
- **precision** – точность (4);
- **recall** – полнота (2);
- **f1_score** – F1 мера (3) **top_3** и **top_5** – меры определяющие вхождение релевантного исполнителя в топ 3 или 5 (5)

Также, если применяется **oversampling**, то разбиение на тренировочную и тестовую выборки происходит следующим образом: разбиваем исходное множество на тренировочное и тестовое, применяем **oversampling** для тренировочного множества и обучаем на нем классификатор, а тестовое множество оставляем неизменным.

На графиках с результатами классификации, используются следующие обозначения:

- по оси *y* следующая структура: используемый метод (соответствует обозначениям из таблицы), выбранные исполнители с количеством заданий, размер тренировочной выборки, размер тестовой выборки;

- По оси x – значение метрики (чем больше, тем лучше).

§6.2.1. Первый уровень исполнения

Разобьем первый уровень исполнения на три подвыборки: в первой будут люди, с количеством выполненных заданий от 10 до 100. Во второй от 100 до 1000. В третьей от 1000 до 6000. В Таблице 3 из Приложения D, и на Рис. 10 - 12 представлены результаты классификации.

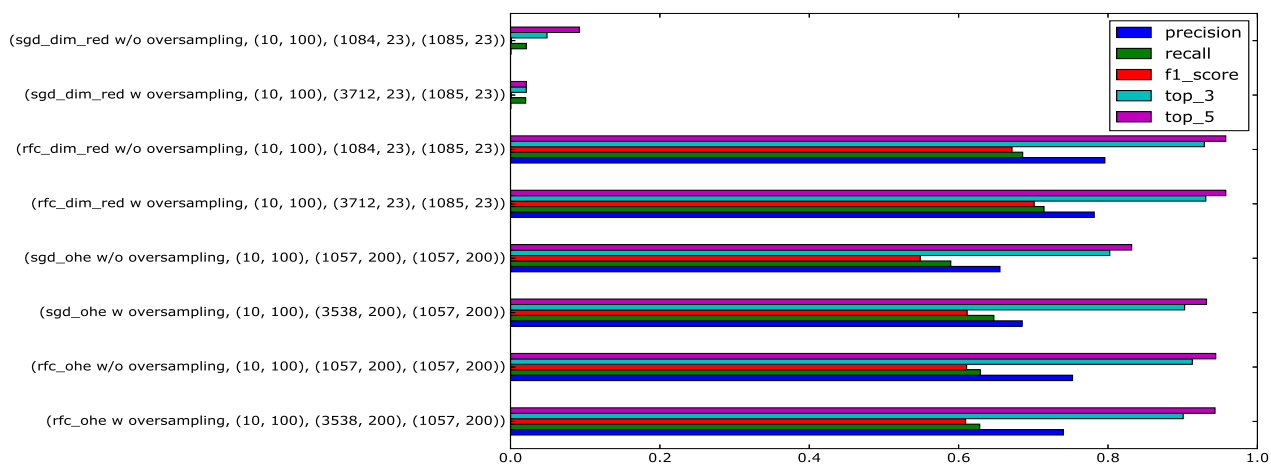


Рис. 10. Результаты классификации на выборке исполнителей с количеством выполненных заданий от 10 до 100.

Количество исполнителей: 64

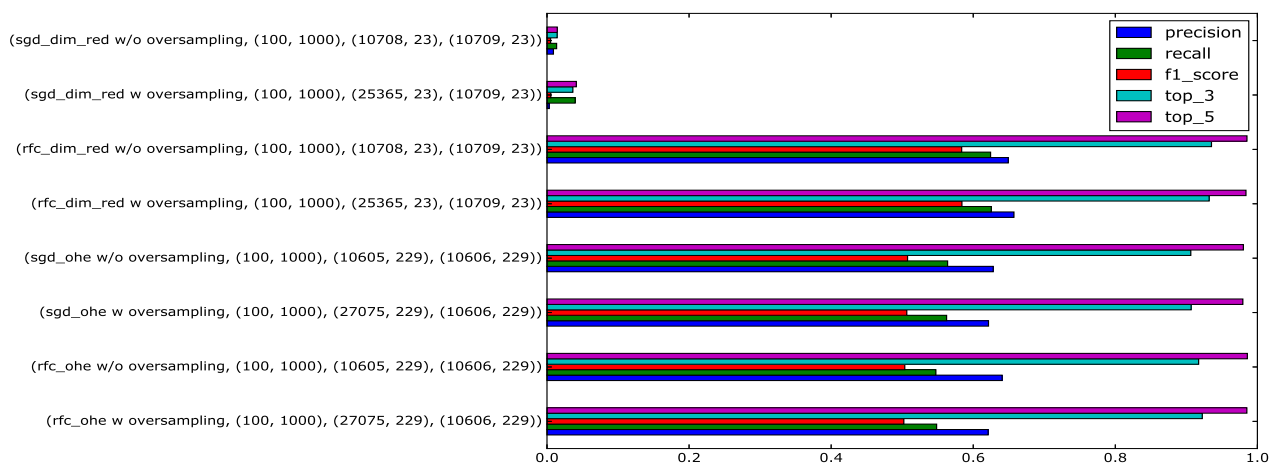


Рис. 11. Результаты классификации на выборке исполнителей с количеством выполненных заданий от 100 до 1000.

Количество исполнителей: 57

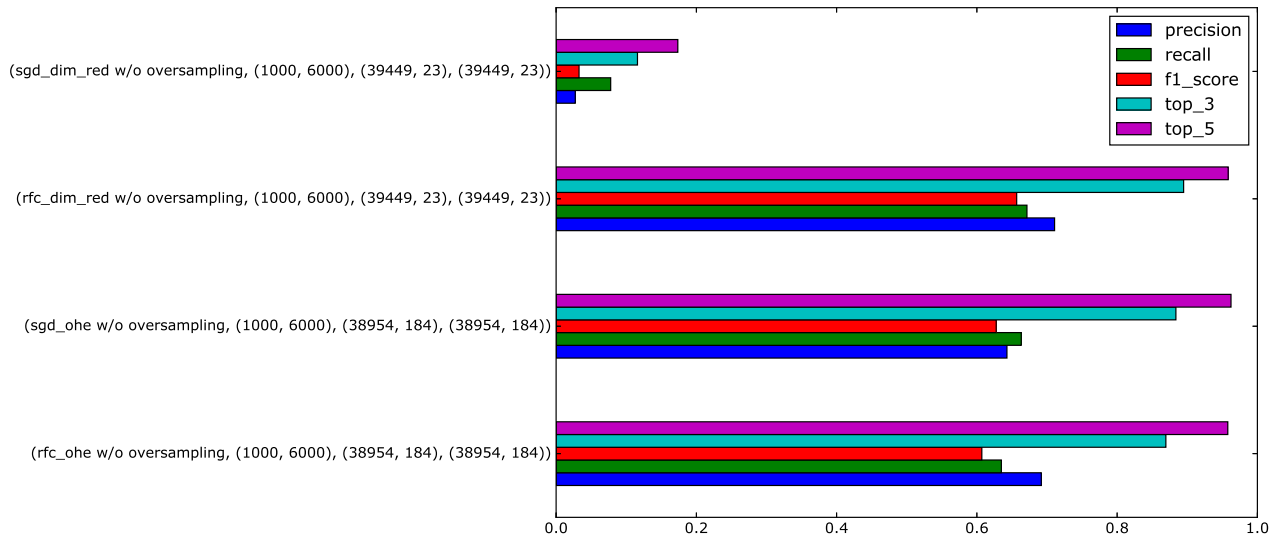


Рис. 12. Результаты классификации на выборке исполнителей с количеством выполненных заданий от 1000 до 6000.

Количество исполнителей: 31

По результатам классификации видно, что метод стохастического градиентного спуска не применим к данным полученным с помощью понижения размерности. Также, на первом уровне исполнения, атрибуты, полученные с помощью понижения размерности, повышают точность классификации.

Рассмотрим теперь, какие атрибуты больше всего влияют на результаты классификации.

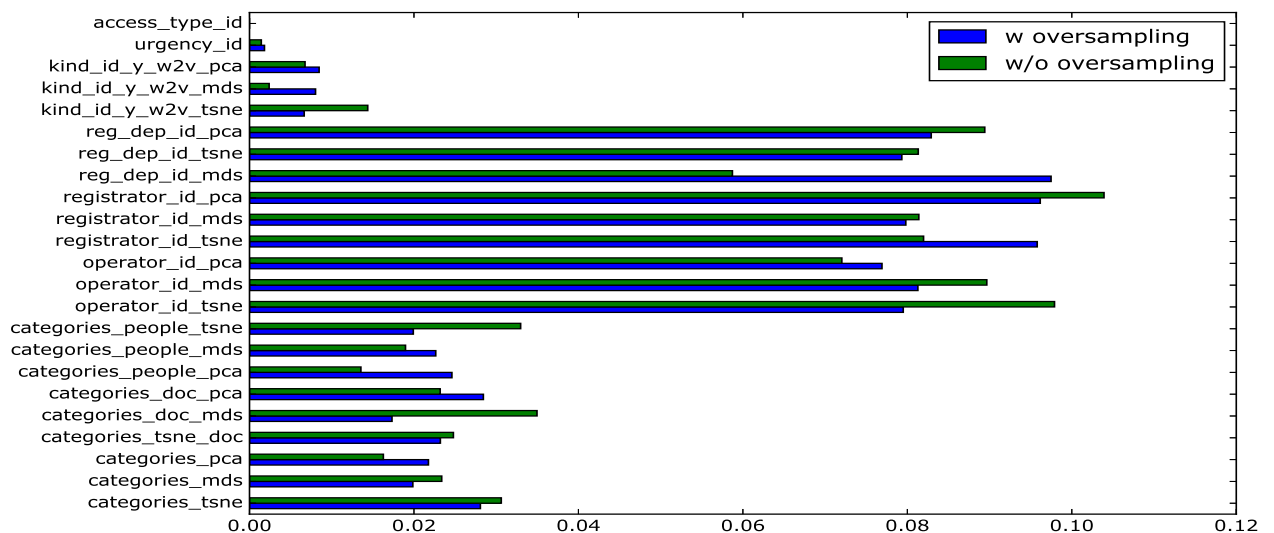


Рис. 13. Важность атрибутов на первом уровне исполнения, с количеством выполненных заданий от 10 до 100

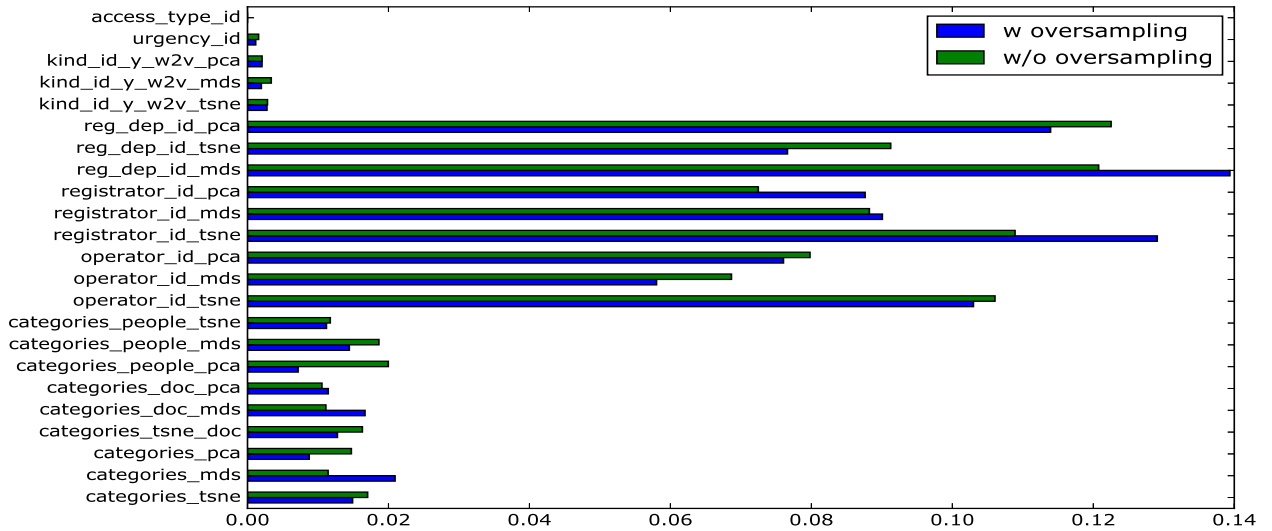


Рис. 14. Важность атрибутов на первом уровне исполнения, с количеством выполненных заданий от 100 до 1000

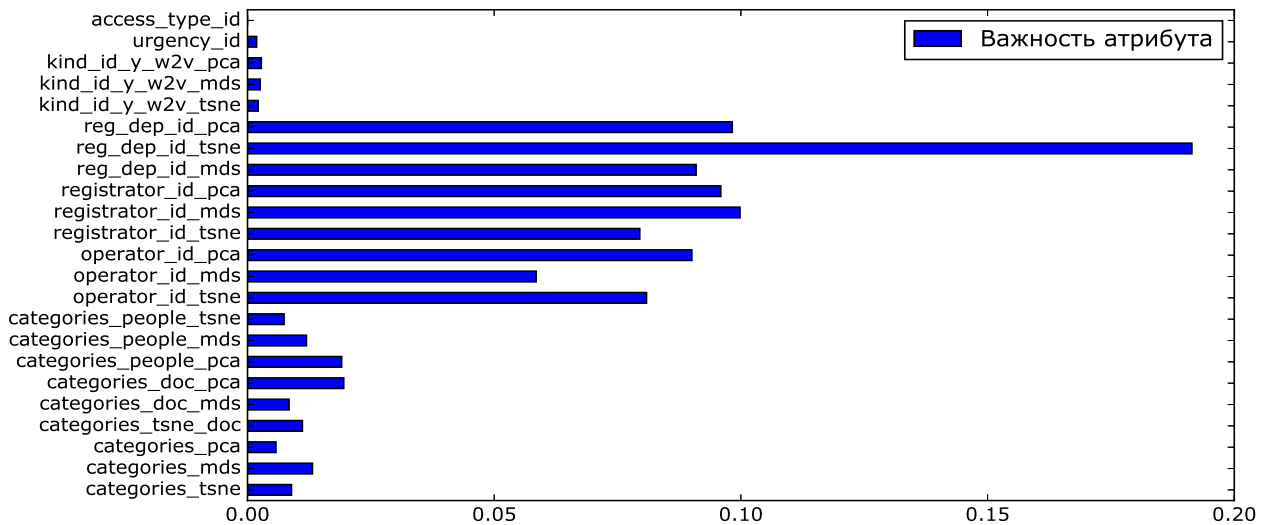


Рис. 15. Важность атрибутов на первом уровне исполнения, с количеством выполненных заданий от 1000 до 6000

На Рис. 13, 14, 15 видно, что oversampling снижает значимость атрибутов, имевших большое значение информативности без oversampling и повышает значимость менее информативных. Также, можно судить, о том, что чем меньше выполнил заданий человек, тем больше оказывают влияние такие атрибуты как *Categories* и *Kind*. Переменные, вроде, *AccessType* и *Urgency* оказались малоинформативными. Зато самое большое влияние оказывают люди, связанные с созданием документа (*Operator*, *Registrator*).

О людях, с низким количеством выполненных заданий, можно судить,

что их еще мало знают и поэтому назначают задания, соответствующие их компетенции. А людям с большим числом заданий, дают более разнообразные задания.

Также можно судить о том, что люди (*Operator, Registrar*) в пространстве большей размерности, имеют линейную структуру, поскольку лучшие результаты дает понижение размерности с помощью метода главных компонент (РСА). А отделы (*RegistrarDepartment*), соответственно, имеют нелинейную структуру.

§6.2.2. Второй уровень исполнения

Для рекомендации исполнителя на втором уровне добавим в атрибуты исполнителя, назначенного на предыдущем уровне. Этот подход можно экстраполировать также и на следующие уровни исполнения, однако в итоге, матрица, полученная с помощью, One hot encoding будет иметь сотни атрибутов. Помимо того, матрица такого размера замедляет вычисления, ее высокая размерность может вести к переобучению.

Разобьем второй уровень исполнения на четыре подвыборки по количеству выполненных заданий: от 10 выполненных заданий до 100, от 100 до 1000, от 100 до 500 и от 1000 до 6000. Для второй подвыборки применим повышение количества примеров, поскольку на этом участке имеется много классов и мало примеров. Тогда получим результаты, представленные в Таблице 4 в Приложении Е и на Рис. 16 – 19.

На графиках видно, что как минимум в 60% случаев в лучших 5 исполнителях будет релевантный сотрудник. Также, как и на первом уровне, SGC плохо обрабатывает данные полученные с помощью понижения размерности.

Низкая точность рекомендации обусловлена тем фактом, что необходимо классифицировать по большому числу классов с малым количеством примеров. Однако, несмотря на это, использование атрибутов, полученных с помощью понижения размерности повышает точность рекомендаций.

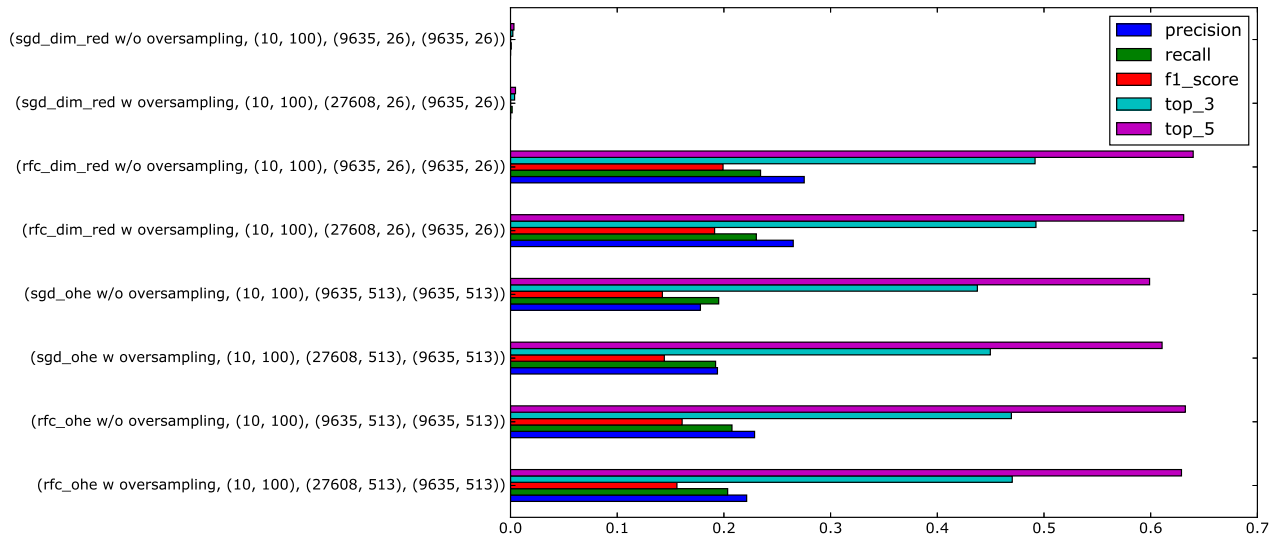


Рис. 16. Результаты классификации на выборке исполнителей с количеством выполненных заданий от 10 до 100.

Количество исполнителей: 493

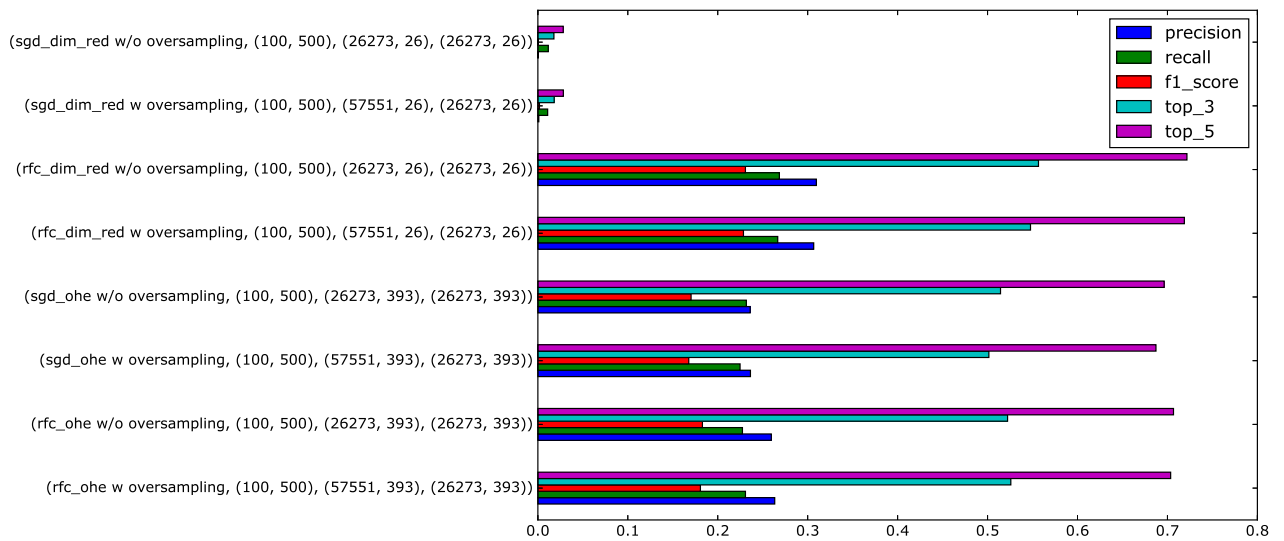


Рис. 17. Результаты классификации на выборке исполнителей с количеством выполненных заданий от 100 до 500.

Количество исполнителей: 233

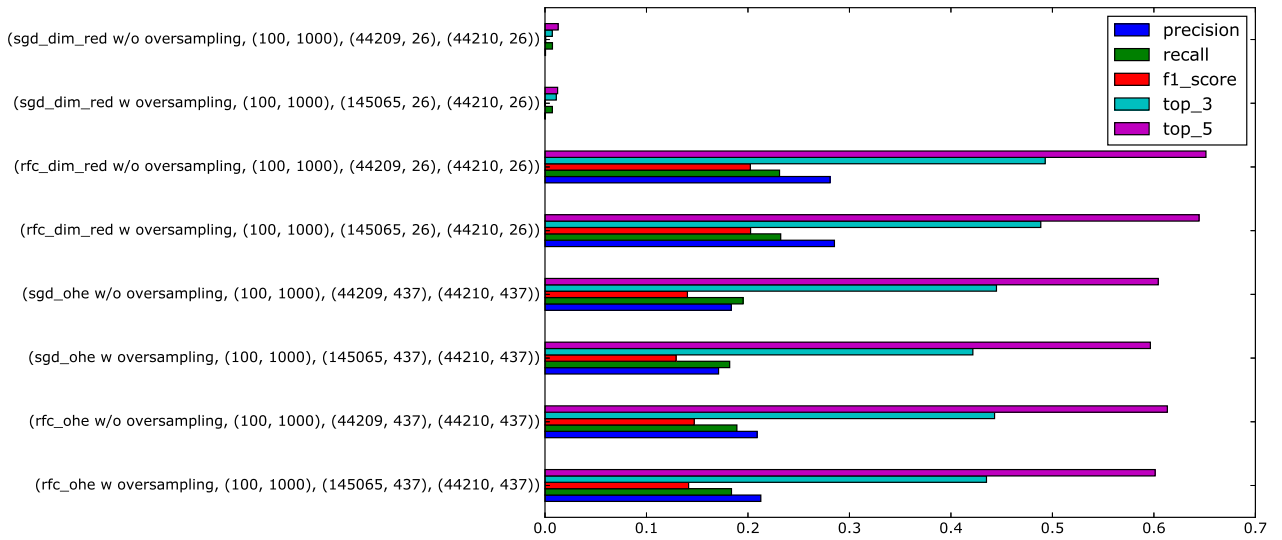


Рис. 18. Результаты классификации на выборке исполнителей с количеством выполненных заданий от 100 до 1000.

Количество исполнителей: 285

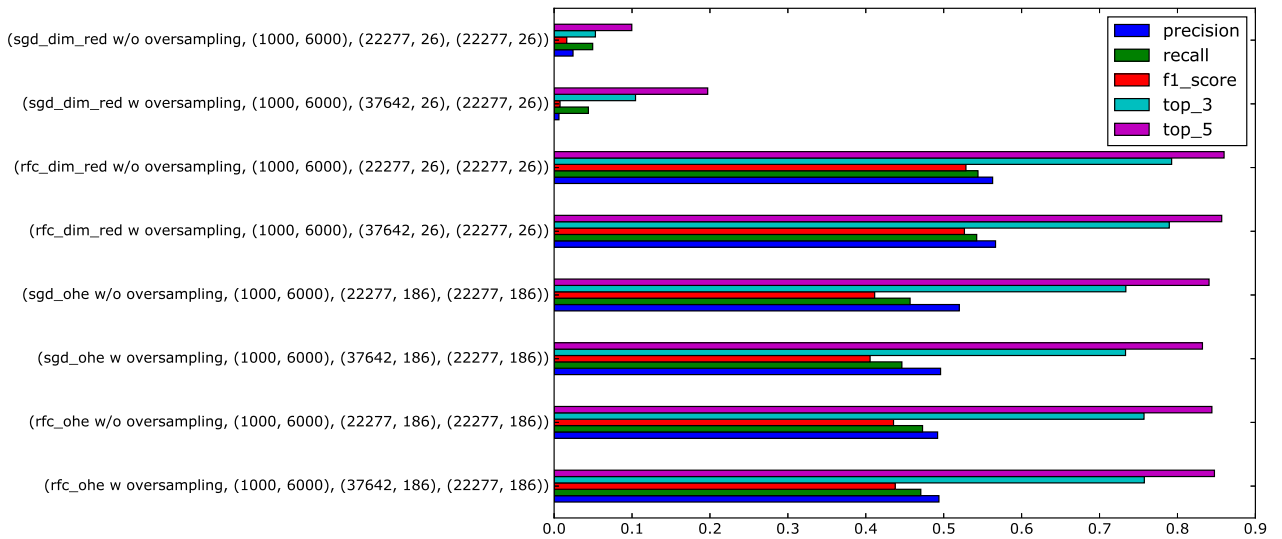


Рис. 19. Результаты классификации на выборке исполнителей с количеством выполненных заданий от 1000 до 6000.

Количество исполнителей: 29

Рассмотрим теперь, какие атрибуты оказывают наибольшее влияние на решение алгоритма. На Рис. 20, 21 видно, что наибольшее влияние оказывает тот человек, который был назначен уровнем выше, тогда как по Рис. 22 переменные, описывающие документ (*Categories, Kind*), оказывают большее влияние.

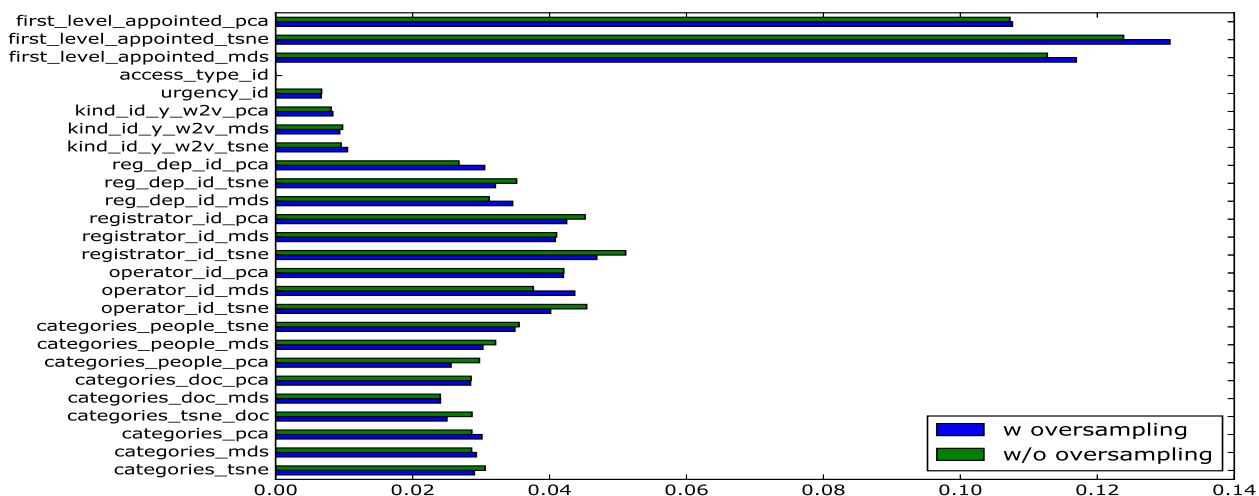


Рис. 20. Важность атрибутов на втором уровне исполнения, с количеством выполненных заданий от 10 до 100

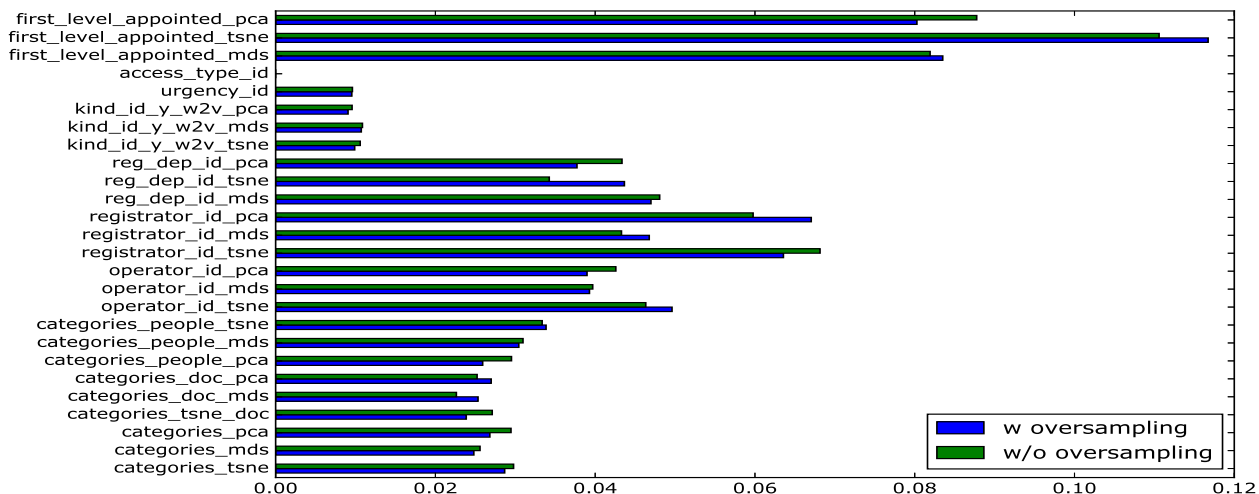


Рис. 21. Важность атрибутов на втором уровне исполнения, с количеством выполненных заданий от 100 до 1000

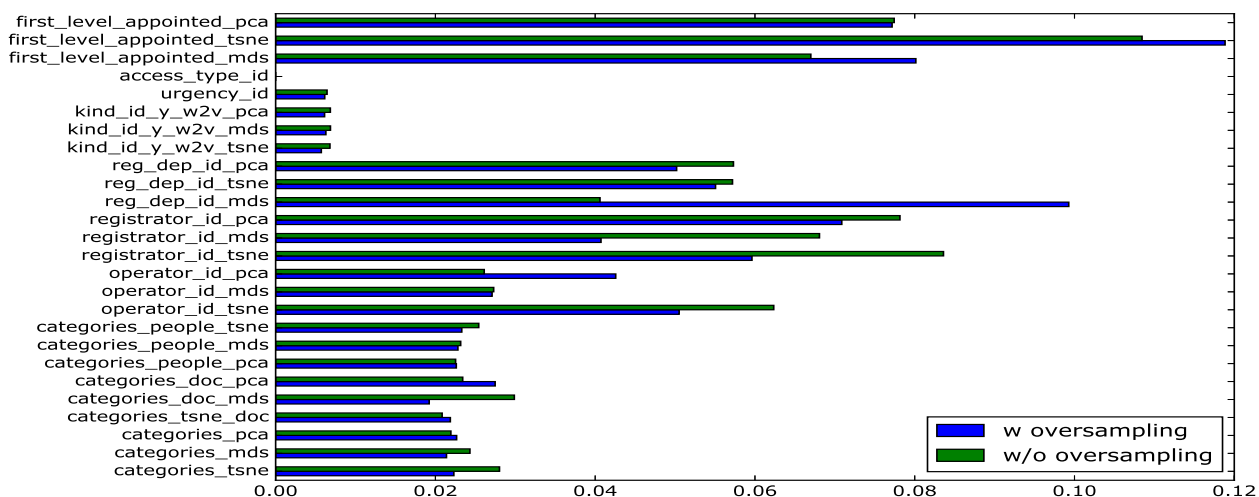


Рис. 22. Важность атрибутов на втором уровне исполнения, с количеством выполненных заданий от 1000 до 6000 с

Oversampling

Oversampling, для исполнителей с малым количеством заданий не вносит значительных изменений в значимости атрибутов, при этом точность рекомендаций в некоторых случаях даже падает.

По получившимся важностям атрибутов можно судить о следующем: исполнитель на втором уровне очень сильно зависит от того, кто был назначен на первом уровне, чем меньше выполнено заданий исполнителем, тем важнее атрибут *first_level_appointed*. Чем больше заданий выполнено, тем больше на равных становятся атрибуты документа и исполнитель первого уровня.

Из атрибутов документа, переменная тип доступа (*AccessType*) не оказывает влияния на решение классификатора. Переменные тип документа (*Kind*) и срочность (*Urgency*) вносят одинаковое количество информации, при этом их важность падает с повышением числа выполненных заданий исполнителем. *Operator*, *Registrator* и *RegistratorDepartment* также несут тем больше информации, чем больше заданий выполнил исполнитель.

Из этого можно сделать вывод, что чем больше исполнитель выполнил заданий на втором уровне, тем больше он похож на исполнителя первого уровня.

§6.2.3. Третий уровень исполнения

На третьем уровне исполнения добавим в атрибуты исполнителей первого и второго уровней. Разобьем этот уровень, по частоте выполнения заданий исполнителями на три подвыборки: от 10 до 100 выполненных заданий, от 100 до 500 и от 100 до 1000.

Получим результаты классификации, представленные на Таблице 5 в Приложении F и графиках 23, 24, 25.

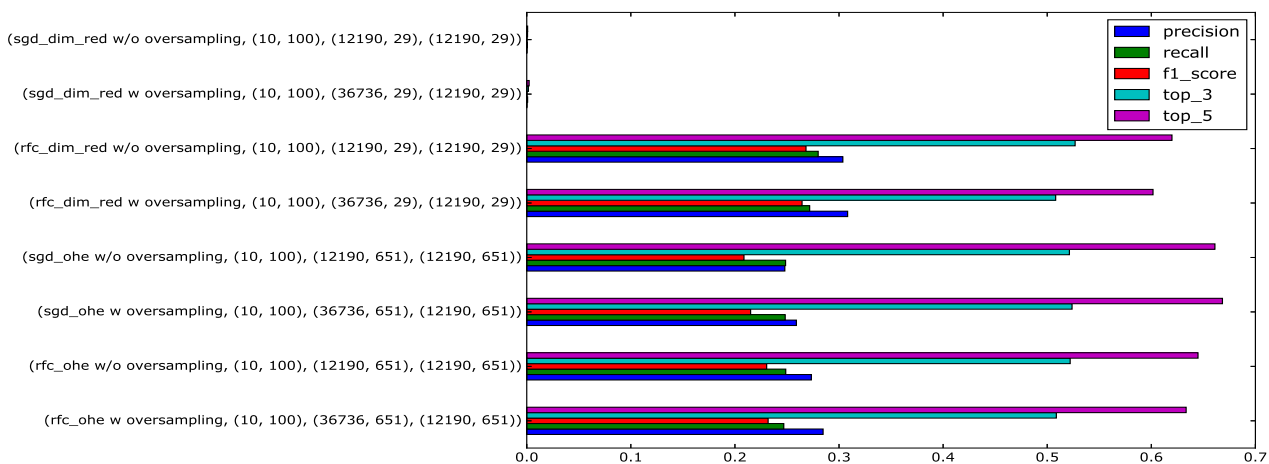


Рис. 23. Результаты классификации на выборке исполнителей с количеством выполненных заданий от 1000 до 6000.

Количество исполнителей: 656

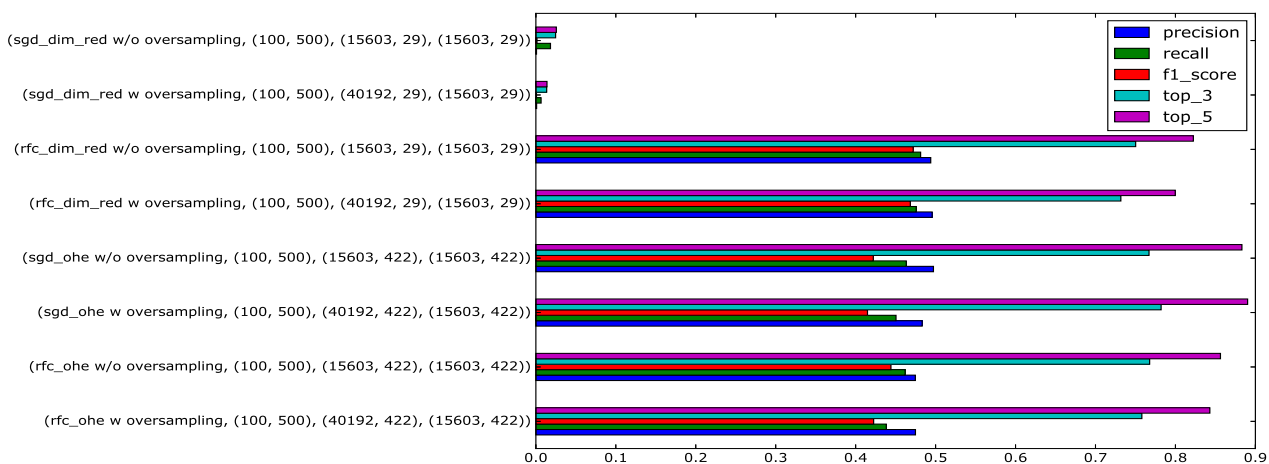


Рис. 24. Результаты классификации на выборке исполнителей с количеством выполненных заданий от 100 до 500.

Количество исполнителей: 157

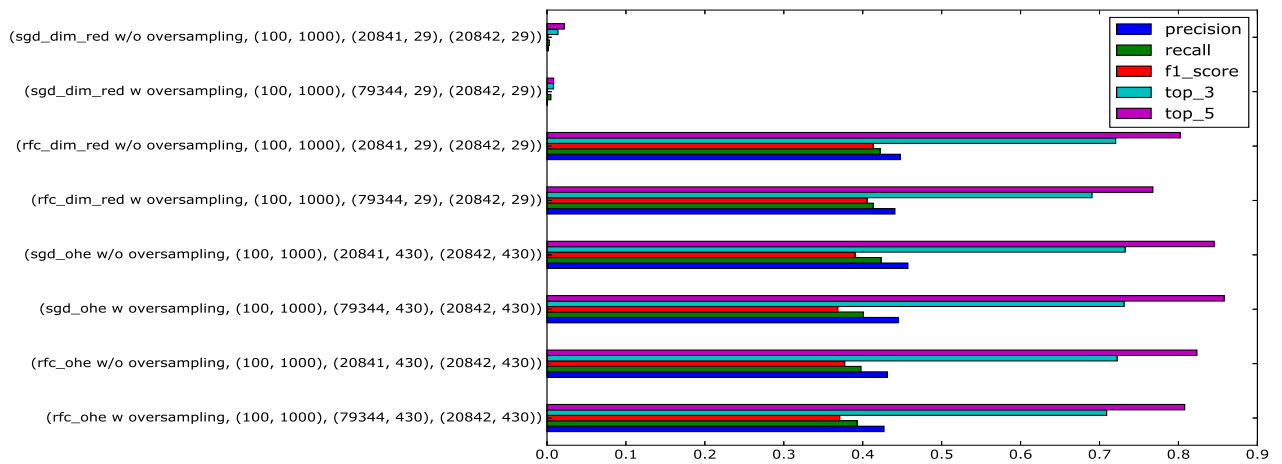


Рис. 25. Результаты классификации на выборке исполнителей с количеством выполненных заданий от 100 до 1000.

Количество исполнителей: 174

На представленных результатах классификации, еще раз убеждаемся, что нельзя использовать данные, полученные с помощью понижения размерности, для классификации с помощью SGC. Также, получаем неплохие результаты классификации для исполнителей с большим количеством выполненных заданий.

Для исполнителей с малым количеством заданий получаются не впечатляющие результаты, однако это также связано с их малым количеством. Обучение классификатора на данных полученных с помощью oversampling незначительно повышает точность рекомендаций, а в некоторых случаях, снижает ее.

Рассмотрим теперь, какие атрибуты оказывают влияние на решение алгоритма. На всех трех графиках 26, 27, 28 видно, что значительную роль играют исполнители первого и второго уровней, а категории документа уходят на второй план.

По графику 26 можно судить, что так как про исполнителя еще известно достаточно мало, то его выбирают в соответствии с документом, при этом, чем больше он выполняет заданий, тем больше исполнители второго уровня выбирают его на задания разного типа.

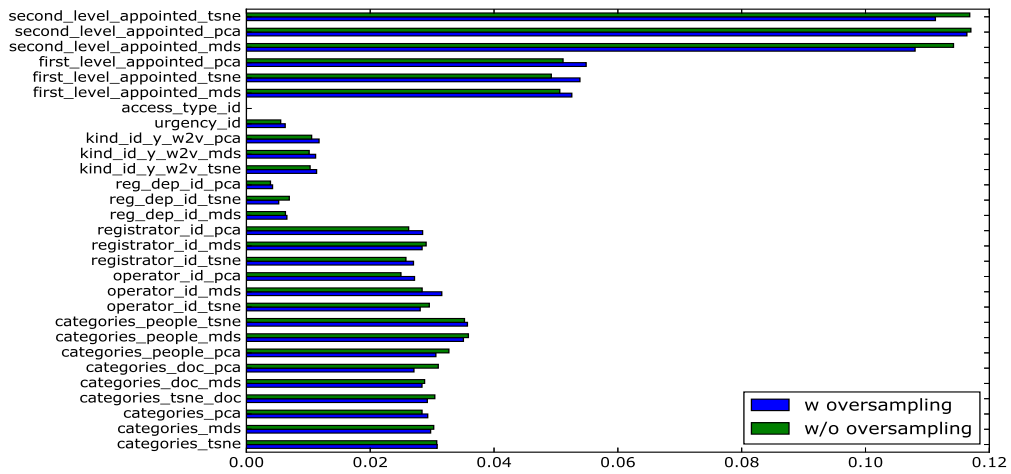


Рис. 26. Важность атрибутов на третьем уровне исполнения, с количеством выполненных заданий от 10 до 100

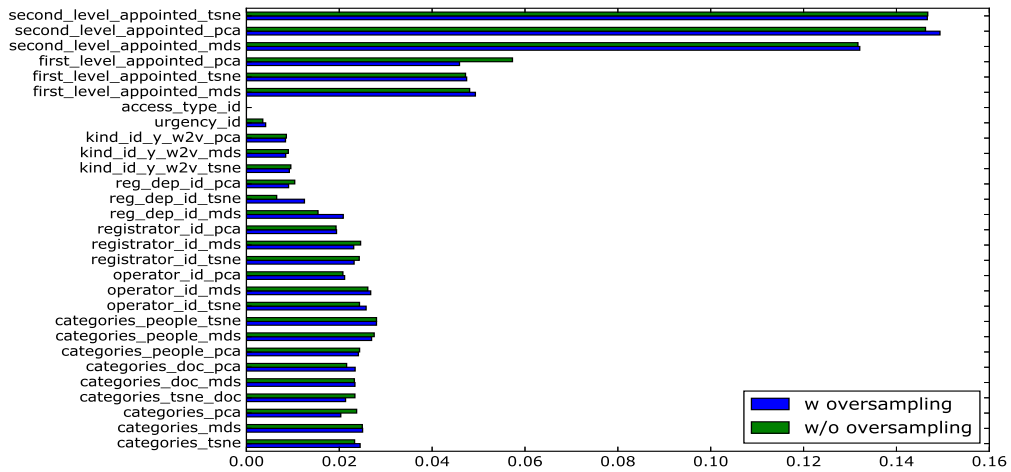


Рис. 27. Важность атрибутов на третьем уровне исполнения, с количеством выполненных заданий от 100 до 500

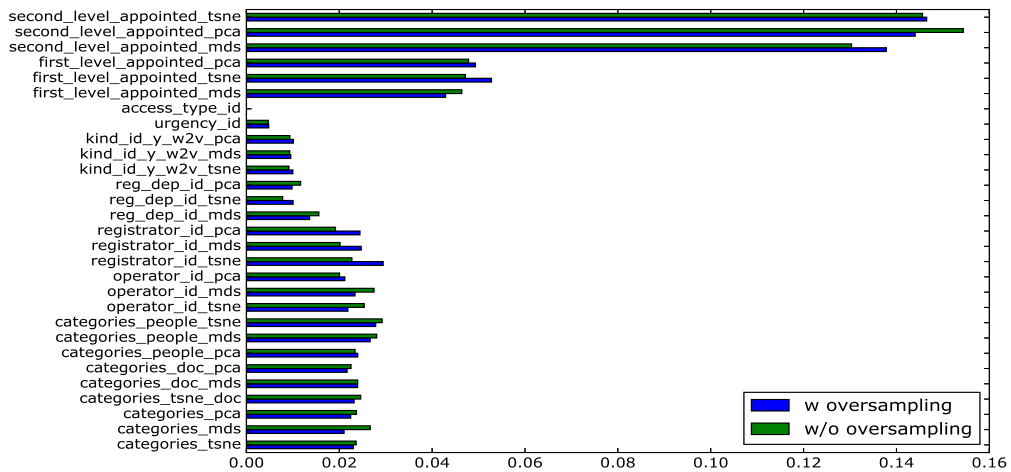


Рис. 28. Важность атрибутов на третьем уровне исполнения, с количеством выполненных заданий от 100 до 1000

Выводы

Экспериментально было доказано, что хоть One hot encoding и является универсальным средством для работы с категориальными переменными, использовать его можно только если необходимо получить результаты быстро, пренебрегая качеством результатов. Если есть время поработать с данными, разобраться в их структуре, то лучше попробовать сохранить их структуру применив подходы представленные в работе. На представленных данных сохранение информации о структуре данных не только давало значительно ниже размерность матрицы X , но также повышало точность прогнозирований.

Заключение

Цель данной работы была в том, чтобы показать, что введение метрик между категориальными переменными и понижение размерности матриц до векторов работает не хуже стандартного подхода One hot encoding. Данная цель была достигнута и более того, данные полученные при использовании рассмотренного подхода не только не ухудшают точность, а повышают ее.

Данная система уже реализована в качестве прототипа в компании Digital Design в коммерческой системе документооборота Docsvision. В качестве дальнейшего развития прототипа рассматривается использование метода построения полного дерева резолуций и расширение системы рекомендаций исполнителей на любых уровнях исполнения. В конечном итоге, стоит задача перевода прототипа в режим коммерческой эксплуатации в качестве дополнительного API системы Docsvision.

Реализация поставленных задач может стать объектом магистерской диссертации.

Список литературы

1. DMS Document management system [Интернет ресурс]: URL:en.wikipedia.org/wiki/Document_management_system (date: 18.03.17)
2. Alkhalaf K. S., Almishal A. I., Almahmoud A. O., Alotaibi M. S. OCR-Based Electronic Documentation Management System // International Journal of Innovation, Management and Technology 2014. Vol. 5, No 5. P. 465–469.
3. Floriana Esposito and Stefano F., Teresa M., Nicola D. Machine Learning for Digital Document Processing: from Layout Analysis to Metadata Extraction // Machine Learning in Document Analysis and Recognition 2008.
4. DigitalDesign [Интернет ресурс]: URL:<http://digdes.ru/about> (дата обращения: 18.03.17)
5. Docsvision [Интернет ресурс]: <http://www.docsvision.com> (дата обращения: 03.05.2017).
6. Manning C. D., Raghavan P. and Schutze H. Introduction to Information Retrieval // Cambridge University Press 2008.
7. Python for Data Analysis: Data Wrangling with Pandas, NumPy, and IPython (01 November 2012) by Wes McKinney
8. kaggle – Сайт с задачами по машинному обучению [Интернет ресурс]: URL: <https://www.kaggle.com>
9. – Решение задачи по прогнозированию цен за дома [Интернет ресурс]: <https://www.kaggle.com/apapiu/regularized-linear-models>
10. Breiman L. Random forests // Machine Learning. 2001. Vol. 45, no. 1. Pp. 5–32.
11. Louppe, G., Wehenkel, L., Suter, A., Geurts, P Understanding variable importances in forests of randomized trees // Advances in Neural Information Processing Systems 26. 2013. C. 431–439.

12. Интерактивное демо создания документа в Docsvision [Интернет ресурс]: URL:<https://marvelapp.com/35d1ihe/screen/13200295>
13. 8 Tactics to Combat Imbalanced Classes in Your Machine Learning Dataset [Интернет ресурс]: URL:goo.gl/IddLHg
14. UUID description on wiki [Интернет ресурс]: URL:https://en.wikipedia.org/wiki/Universally_unique_identifier (дата обращения: 15.05.2017).
15. Random forest на wikipedia [Интернет ресурс]: URL:https://en.wikipedia.org/wiki/Random_forest (дата обращения: 18.03.17).
16. Композиции алгоритмов, основанные на случайном лесе // http://www.machinelearning.ru/wiki/images/d/d8/2015_517_RyzhkovAM.pdf URL:http://www.machinelearning.ru/wiki/images/d/d8/2015_517_RyzhkovAM.pdf (дата обращения: 03.05.2017).
17. Sklearn [Интернет ресурс]: URL:<http://scikit-learn.org> (дата обращения: 15.05.2017).
18. Stochastic Gradient Descent // <http://scikit-learn.org/stable/modules/sgd.html> URL:<http://scikit-learn.org/stable/modules/sgd.html> (дата обращения: 15.05.2017).
19. Large-Scale Machine Learning with Stochastic Gradient Descent In Proceedings of the 19th International Conference on Computational Statistics (COMPSTAT'2010) (August 2010), pp. 177-187 by Luon Bottou edited by Yves Lechevallier, Gilbert Saporta
20. Tranmer M., Elliot M. Multiple linear regression //The Cathie Marsh Centre for Census and Survey Research (CCSR). Ц 2008.
21. Radim R., Sojka P. Software Framework for Topic Modelling with Large Corpora // Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks. Valletta, Malta: ELRA, 2010. С. 45–50.

22. Word2Vec в примерах [Интернет ресурс] URL:<https://habrahabr.ru/post/249215/> (дата обращения: 15.05.2017).
23. Mikolov, T., Sutskever, I., Chen, K., Greg S. C., Dean, J. Distributed Representations of Words and
24. Правительство Российской Федерации [Интернет ресурс]: <http://government.ru> (дата обращения: 15.05.2017).
25. Обученная word2vec модель и документы с сайта правительства на лето 2016 года [Интернет ресурс]: URL:<https://yadi.sk/d/d0GUKF9R3Hs9WT> (дата обращения: 15.05.2017).
26. Лекция 13. Уменьшение размерности в данных. Метод главных компонент. [Интернет ресурс]: URL:<http://www.machinelearning.ru/wiki/images/4/45/SMAIS2009-13.pdf> (дата обращения: 15.05.2017).
27. Pearson K., On lines and planes of closest fit to systems of points in space, Philosophical Magazine, (1901) 2, 559-572
28. L.J.P. van der Maaten and G.E. Hinton. Visualizing High-Dimensional Data Using t-SNE. Journal of Machine Learning Research 9(Nov):2579-2605, 2008.
29. Классификация секретной информации в РФ [Интернет ресурс]: URL:https://ru.wikipedia.org/wiki/Классификация_секретной_информации_в_России (дата обращения: 15.05.2017).

Приложение

Приложение А. Пример document.json файла

```
1 {
2   "Type": 0,
3   "Content": "sample string 1",
4   "RegistrationDate": "2017-04-20T15:28:37.6113165+03:00",
5   "CompletionDate": "2017-04-20T15:28:37.6113165+03:00",
6   "Kind": {
7     "Name": "sample string 3",
8     "Id": "55e8453b-9cb8-4679-b56a-f39d4f534d52"
9   },
10  "AccessType": {
11    "Name": "sample string 3",
12    "Id": "e578eb30-32cb-4a78-9a0e-a473da3f714a"
13  },
14  "Urgency": {
15    "Name": "sample string 1",
16    "Id": "612d1a9b-8e45-4b9a-8cbc-dbcef4530cb9"
17  },
18  "Operator": {
19    "Type": 0,
20    "Position": "sample string 5",
21    "PositionImportance": 1,
22    "Status": 0,
23    "Manager": {
24      "Id": "ffd40955-ad37-4a02-8e88-dd455fed3f5b"
25    },
26    "Importance": 1,
27    "Unit": {
28      "Type": 1,
29      "UnitType": 6,
30      "Name": "sample string 7",
31      "Id": "1df592ea-6695-41b0-893e-7b3d2cf2a303"
32    },
33    "Gender": 0,
34    "Name": "sample string 12",
35    "Id": "f02be821-f55e-4de6-b15f-ad2ae25135d5"
36  },
37  "RegistrarDepartment": {
38    "Type": 1,
39    "Manager": {
40      "Type": 0,
41      "Position": "sample string 5",
42      "PositionImportance": 1,
43      "Status": 0,
44      "Manager": {
45        "Id": "ffd40955-ad37-4a02-8e88-dd455fed3f5b"
46      },
47      "Importance": 1,
```

```

48     "Gender": 0,
49     "Name": "sample string 12",
50     "Id": "f02be821-f55e-4de6-b15f-ad2ae25135d5"
51 },
52 "UnitType": 6,
53 "Name": "sample string 7",
54 "Id": "1df592ea-6695-41b0-893e-7b3d2cf2a303"
55 },
56 "Registrator": {
57     "Type": 0,
58     "Position": "sample string 5",
59     "PositionImportance": 1,
60     "Status": 0,
61     "Manager": {
62         "Id": "ffd40955-ad37-4a02-8e88-dd455fed3f5b"
63     },
64     "Importance": 1,
65     "Unit": {
66         "Type": 1,
67         "UnitType": 6,
68         "Name": "sample string 7",
69         "Id": "1df592ea-6695-41b0-893e-7b3d2cf2a303"
70     },
71     "Gender": 0,
72     "Name": "sample string 12",
73     "Id": "f02be821-f55e-4de6-b15f-ad2ae25135d5"
74 },
75 "ExternalControl": true,
76 "TemplateId": "ff35952c-e21d-4905-b7b8-10efd354304e",
77 "Performer": {
78     "Type": 0,
79     "Position": "sample string 5",
80     "PositionImportance": 1,
81     "Status": 0,
82     "Manager": {
83         "Id": "ffd40955-ad37-4a02-8e88-dd455fed3f5b"
84     },
85     "Importance": 1,
86     "Unit": {
87         "Type": 1,
88         "UnitType": 6,
89         "Name": "sample string 7",
90         "Id": "1df592ea-6695-41b0-893e-7b3d2cf2a303"
91     },
92     "Gender": 0,
93     "Name": "sample string 12",
94     "Id": "f02be821-f55e-4de6-b15f-ad2ae25135d5"
95 },
96 "Files": [
97     {
98         "Created": "2017-04-20T15:28:37.6113165+03:00",
99         "Changed": "2017-04-20T15:28:37.6113165+03:00",
100        "Extension": "sample string 3",
101        "Size": 4,
102        "Name": "sample string 6",

```

```

103     "OwnerId": "e54e87b4-5762-43e0-81a4-e72e45d1e11f",
104     "Importance": 8,,
105     "ParentId": "692f7ff5-8bbd-4b2c-9c80-3603740974bd",
106     "Id": "e185baa1-dfdb-42ef-b324-68124fc03a2e"
107   }],
108   "ParentId": "692f7ff5-8bbd-4b2c-9c80-3603740974bd",
109   "Id": "e185baa1-dfdb-42ef-b324-68124fc03a2e"
110 }
111 ],
112 "Addressees": [
113   {
114     "OrderNumber": 1,
115     "Reference": {
116       "Id": "ffd40955-ad37-4a02-8e88-dd455fed3f5b"
117     },
118     "Type": 0,
119     "AddresseeType": 0,
120     "OutgoingNumber": "sample string 1",
121     "OutRegDate": "2017-04-20T15:28:37.6113165+03:00",
122     "OutSignedBy": {
123       "Address": "sample string 8",
124       "Name": "sample string 9",
125       "Id": "caf29ff2-e6e7-4bb7-9baf-71d21fa5ff5d"
126     },
127     "SendTo": {
128       "Address": "sample string 8",
129       "Name": "sample string 9",
130       "Id": "caf29ff2-e6e7-4bb7-9baf-71d21fa5ff5d"
131     },
132     "NumberOfCopies": "sample string 4",
133     "PageCount": "sample string 5",
134     "TaskId": {
135       "Id": "ffd40955-ad37-4a02-8e88-dd455fed3f5b"
136     },
137     "Id": "3ba56ffe-eefb-492d-8d6c-8209d108321c"
138   }],
139 "Executors": [
140   {
141     "Order": 1,
142     "Employee": {
143       "Type": 0,
144       "Position": "sample string 5",
145       "PositionImportance": 1,
146       "Status": 0,
147       "Manager": {
148         "Id": "ffd40955-ad37-4a02-8e88-dd455fed3f5b"
149       },
150       "Importance": 1,
151       "Unit": {
152         "Type": 1,
153         "UnitType": 6,
154         "Name": "sample string 7",
155         "Id": "1df592ea-6695-41b0-893e-7b3d2cf2a303"
156       },
157       "Gender": 0,

```



```

158     "Name": "sample string 12",
159     "Id": "f02be821-f55e-4de6-b15f-ad2ae25135d5"
160   },
161   "Id": "96ead854-9ae6-43ad-be79-636a49e75d70"
162  }],
163  "Approvers": [
164    {
165      "Approver": {
166        "Type": 0,
167        "Position": "sample string 5",
168        "PositionImportance": 1,
169        "Status": 0,
170        "Manager": {
171          "Id": "ffd40955-ad37-4a02-8e88-dd455fed3f5b"
172        },
173        "Importance": 1,
174        "Unit": {
175          "Type": 1,
176          "UnitType": 6,
177          "Name": "sample string 7",
178          "Id": "1df592ea-6695-41b0-893e-7b3d2cf2a303"
179        },
180        "Gender": 0,
181        "Name": "sample string 12",
182        "Id": "f02be821-f55e-4de6-b15f-ad2ae25135d5"
183      },
184      "ApprovalDate": "2017-04-20T15:28:37.6113165+03:00",
185      "IsApproved": true,
186      "ApprovalStatus": 0,
187      "Id": "6b52896e-15af-455b-9e04-6d20b2cd5bdf"
188    }
189  ],
190  "NumberOfCopies": "sample string 5",
191  "NumberOfPages": "sample string 6",
192  "Categories": [
193    {
194      "Name": "sample string 1",
195      "Id": "612d1a9b-8e45-4b9a-8cbc-dbcef4530cb9"
196    }
197  ],
198  "Description": "sample string 7",
199  "State": {
200    "Name": "sample string 1",
201    "Id": "4007f36e-a4a0-4715-b714-943d0c06c8af"
202  },
203  "IsTemplate": true,
204  "CreationDate": "2017-04-20T15:28:37.6113165+03:00",
205  "ChangeDate": "2017-04-20T15:28:37.6113165+03:00",
206  "Url": "sample string 11",
207  "Id": "fa9fbe3e-2d75-4a91-b9ed-e3482a207832"
208 }

```

Приложение В. Пример resolution.json файла

```

1  {
2  "Tasks": [
3    {
4      "Parts": [
5        {
6          "Executors": [
7            {
8              "Id": "64bdca00-16a0-49ca-825d-a3f6501b2201",
9              "PartId": "86f69674-e474-45fd-b048-91c6a5f30a1d",
10             "ExecutorNumber": 3,
11             "ExecutorType": 0,
12             "Executor": {
13               "Name": "sample string 1",
14               "Id": "0481194f-3de2-45bc-8930-f88bf0bf1e1f"
15             },
16             "ExecutorKind": 0,
17             "Tasks": [],
18             "ReportRequire": true,
19             "AutoPerform": true,
20             "PursuanceRequire": true,
21             "FirstResponsible": true,
22             "ApprovalStatus": 0
23           }
24         ],
25         "Id": "db5c83a2-00ac-4750-8f14-84f6e35faf6a",
26         "TaskId": "af468446-752d-407a-8cf0-4f4b8e425c9b",
27         "Cancelled": true,
28         "ControlTerm": "2017-04-20T15:42:46.5027643+03:00",
29         "PartText": "sample string 7",
30         "Urgency": {
31           "Name": "sample string 1",
32           "Id": "0481194f-3de2-45bc-8930-f88bf0bf1e1f"
33         },
34         "Categories": [
35           {
36             "Name": "sample string 1",
37             "Id": "0481194f-3de2-45bc-8930-f88bf0bf1e1f"
38           }
39         ],
40       ],
41       "Description": "sample string 1",
42       "Id": "c65ade91-6424-406a-af6e-2edb26ff9f88",
43       "ApprovalStatus": 0,
44       "Kind": {
45         "Name": "sample string 3",
46         "Id": "3d5e9360-92e4-4fc3-9638-aade10795dce"
47       },
48       "State": {
49         "Name": "sample string 1",
50         "Id": "bd67965e-5b17-48ed-8895-116d3ed696ee"
51       },
52       "AppointedType": 0,
53       "ExecutesDisplayName": "sample string 4",
54       "AuthorDisplayName": "sample string 5",
55       "AppointedDisplayName": "sample string 6",

```

```

56     "CreationDateTime": "2017-04-20T15:42:46.518389+03:00",
57     "FinishworkTerm": "2017-04-20T15:42:46.518389+03:00",
58     "ControlTerm": "2017-04-20T15:42:46.518389+03:00",
59     "ActualTerm": "2017-04-20T15:42:46.518389+03:00",
60     "ApprovalDate": "2017-04-20T15:42:46.518389+03:00",
61     "Executes": {
62         "Type": 0,
63         "Position": "sample string 5",
64         "PositionImportance": 1,
65         "Status": 0,
66         "Manager": {
67             "Id": "246f38bb-8be9-4d4d-aa03-9011fa3bf23f"
68         },
69         "Importance": 1,
70         "Unit": {
71             "Type": 1,
72             "UnitType": 6,
73             "Name": "sample string 7",
74             "Id": "f661b40c-4867-48aa-9bb6-19d576f38f52"
75         },
76         "Gender": 0,
77         "Name": "sample string 12",
78         "Id": "200e825a-796b-4580-bb3d-417e7676d2fa"
79     },
80     "Appointed": {
81         "Type": 0,
82         "Position": "sample string 5",
83         "PositionImportance": 1,
84         "Status": 0,
85         "Manager": {
86             "Id": "246f38bb-8be9-4d4d-aa03-9011fa3bf23f"
87         },
88         "Importance": 1,
89         "Unit": {
90             "Type": 1,
91             "UnitType": 6,
92             "Name": "sample string 7",
93             "Id": "f661b40c-4867-48aa-9bb6-19d576f38f52"
94         },
95         "Gender": 0,
96         "Name": "sample string 12",
97         "Id": "200e825a-796b-4580-bb3d-417e7676d2fa"
98     },
99     "Author": {
100         "Type": 0,
101         "Position": "sample string 5",
102         "PositionImportance": 1,
103         "Status": 0,
104         "Manager": {
105             "Id": "246f38bb-8be9-4d4d-aa03-9011fa3bf23f"
106         },
107         "Importance": 1,
108         "Unit": {
109             "Type": 1,
110             "UnitType": 6,

```

```
111         "Name": "sample string 7",
112         "Id": "f661b40c-4867-48aa-9bb6-19d576f38f52"
113     },
114     "Gender": 0,
115     "Name": "sample string 12",
116     "Id": "200e825a-796b-4580-bb3d-417e7676d2fa"
117 },
118 "PerformingReport": true,
119 "HasFiles": true,
120 "ApproverOrder": 19
121 }
122 ],
123 "Id": "cc6cbb6a-7142-4622-9d9a-a14c20e111aa",
124 "Description": "sample string 2",
125 "CreationDateTime": "2017-04-20T15:42:46.518389+03:00",
126 }
```

Приложение С. Восемь уровней исполнения

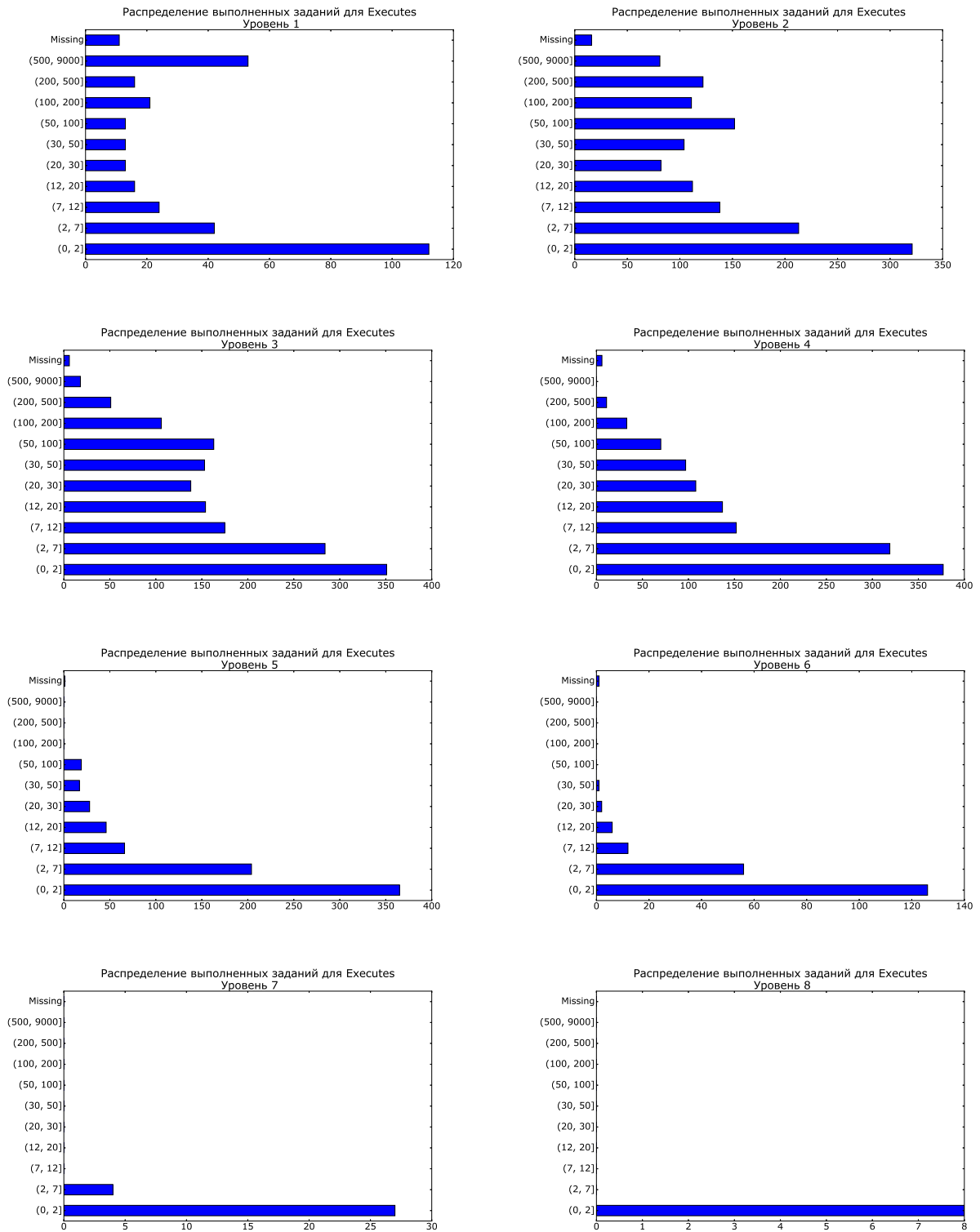


Рис. 29. По оси Y интервалы с количеством выполненных заданий, а по X сколько исполнителей заданий (*Appointed*) в каждой корзине

Приложение D. Результаты классификации на первом уровне исполнения

subset	train_size	test_size	method	precision	recall	f1	top_3	top_5	n_classes
(10, 100)	(3712, 23)	(1085, 23)	RFC_dim_red w oversampling	0.81	0.68	0.67	0.92	0.95	64
(10, 100)	(1084, 23)	(1085, 23)	RFC_dim_red w/o oversampling	0.80	0.69	0.68	0.95	0.96	64
(10, 100)	(3712, 23)	(1085, 23)	SGD_dim_red w oversampling	0.00	0.00	0.00	0.00	0.00	64
(10, 100)	(1084, 23)	(1085, 23)	SGD_dim_red w/o oversampling	0.0	0.01	0.0	0.01	0.01	64
(10, 100)	(3538, 200)	(1057, 200)	RFC_ohc w oversampling	0.74	0.63	0.61	0.90	0.94	64
(10, 100)	(1057, 200)	(1057, 200)	RFC_ohc w/o oversampling	0.75	0.63	0.61	0.91	0.94	64
(10, 100)	(3538, 200)	(1057, 200)	SGD_ohc w oversampling	0.69	0.65	0.61	0.90	0.93	64
(10, 100)	(1057, 200)	(1057, 200)	SGD_ohc w/o oversampling	0.66	0.59	0.55	0.80	0.83	64
(100, 500)	(7844, 23)	(4077, 23)	RFC_dim_red w oversampling	0.74	0.73	0.70	0.95	0.99	37
(100, 500)	(4077, 23)	(4077, 23)	RFC_dim_red w/o oversampling	0.72	0.73	0.70	0.95	0.99	37
(100, 500)	(7844, 23)	(4077, 23)	SGD_dim_red w oversampling	0.04	0.05	0.03	0.06	0.13	37
(100, 500)	(4077, 23)	(4077, 23)	SGD_dim_red w/o oversampling	0.00	0.01	0.00	0.01	0.02	37
(100, 500)	(8103, 171)	(4031, 171)	RFC_ohc w oversampling	0.64	0.67	0.62	0.92	0.99	37
(100, 500)	(4031, 171)	(4031, 171)	RFC_ohc w/o oversampling	0.65	0.67	0.62	0.92	0.99	37
(100, 500)	(8103, 171)	(4031, 171)	SGD_ohc w oversampling	0.63	0.65	0.61	0.92	0.99	37
(100, 500)	(4031, 171)	(4031, 171)	SGD_ohc w/o oversampling	0.54	0.62	0.54	0.88	0.96	37
(100, 1000)	(25365, 23)	(10709, 23)	RFC_dim_red w oversampling	0.66	0.63	0.59	0.94	0.98	57
(100, 1000)	(25365, 23)	(10709, 23)	SGD_dim_red w oversampling	0.02	0.06	0.01	0.03	0.05	57
(100, 1000)	(10708, 23)	(10709, 23)	SGD_dim_red w/o oversampling	0.00	0.01	0.00	0.07	0.10	57
(100, 1000)	(27075, 229)	(10606, 229)	RFC_ohc w oversampling	0.62	0.55	0.50	0.92	0.99	57
(100, 1000)	(10605, 229)	(10606, 229)	RFC_ohc w/o oversampling	0.64	0.55	0.50	0.92	0.99	57
(100, 1000)	(27075, 229)	(10606, 229)	SGD_ohc w oversampling	0.62	0.56	0.51	0.91	0.98	57
(100, 1000)	(10605, 229)	(10606, 229)	SGD_ohc w/o oversampling	0.63	0.56	0.51	0.91	0.98	57
(1000, 6000)	(39449, 23)	(39449, 23)	RFC_dim_red w/o oversampling	0.70	0.67	0.66	0.89	0.96	31
(1000, 6000)	(39449, 23)	(39449, 23)	SGD_dim_red w/o oversampling	0.02	0.05	0.02	0.08	0.09	31
(1000, 6000)	(38954, 184)	(38954, 184)	RFC_ohc w/o oversampling	0.69	0.63	0.61	0.87	0.96	31
(1000, 6000)	(38954, 184)	(38954, 184)	SGD_ohc w/o oversampling	0.64	0.66	0.63	0.88	0.96	31

Таблица 3. Результаты классификации на первом уровне исполнения. test_size=0.5.

Приложение Е. Результаты классификации на Втором уровне исполнения

subset	train_size	test_size	method	precision	recall	f1	top_3	top_5	n_classes
(10, 100)	(27608, 513)	(9635, 513)	RFC_ohc w oversampling	0.22	0.20	0.16	0.47	0.63	493
(10, 100)	(9635, 513)	(9635, 513)	RFC_ohc w/o oversampling	0.23	0.21	0.16	0.47	0.63	493
(10, 100)	(27608, 513)	(9635, 513)	RFC_ohc w oversampling	0.19	0.19	0.14	0.45	0.61	493
(10, 100)	(9635, 513)	(9635, 513)	SGD_ohc w/o oversampling	0.18	0.20	0.14	0.44	0.60	493
(10, 100)	(27608, 26)	(9635, 26)	RFC_dim_red w oversampling	0.27	0.23	0.19	0.49	0.63	493
(10, 100)	(9635, 26)	(9635, 26)	RFC_dim_red w/o oversampling	0.28	0.23	0.20	0.49	0.64	493
(10, 100)	(27608, 26)	(9635, 26)	SGD_ohc w oversampling	0.00	0.00	0.00	0.00	0.00	493
(10, 100)	(9635, 26)	(9635, 26)	SGD_ohc w/o oversampling	0.00	0.00	0.00	0.00	0.00	493
(100, 500)	(57551, 393)	(26273, 393)	RFC_ohc w oversampling	0.26	0.23	0.18	0.53	0.70	233
(100, 500)	(26273, 393)	(26273, 393)	RFC_ohc w/o oversampling	0.26	0.23	0.18	0.52	0.71	233
(100, 500)	(57551, 393)	(26273, 393)	RFC_ohc w oversampling	0.24	0.22	0.17	0.50	0.69	233
(100, 500)	(26273, 393)	(26273, 393)	SGD_ohc w/o oversampling	0.24	0.23	0.17	0.51	0.70	233
(100, 500)	(57551, 26)	(26273, 26)	RFC_dim_red w oversampling	0.31	0.27	0.23	0.55	0.72	233
(100, 500)	(26273, 26)	(26273, 26)	RFC_dim_red w/o oversampling	0.31	0.27	0.23	0.56	0.72	233
(100, 500)	(57551, 26)	(26273, 26)	SGD_dim_red w oversampling	0.00	0.01	0.00	0.02	0.03	233
(100, 500)	(26273, 26)	(26273, 26)	SGD_dim_red w/o oversampling	0.00	0.01	0.00	0.02	0.03	233
(100, 1000)	(145065, 437)	(44210, 437)	RFC_ohc w oversampling	0.21	0.18	0.14	0.44	0.60	285
(100, 1000)	(44209, 437)	(44210, 437)	RFC_ohc w/o oversampling	0.21	0.19	0.15	0.44	0.61	285
(100, 1000)	(145065, 437)	(44210, 437)	RFC_ohc w oversampling	0.17	0.18	0.13	0.42	0.60	285
(100, 1000)	(44209, 437)	(44210, 437)	SGD_ohc w/o oversampling	0.18	0.20	0.14	0.44	0.60	285
(100, 1000)	(145065, 26)	(44210, 26)	RFC_dim_red w oversampling	0.29	0.23	0.20	0.49	0.64	285
(100, 1000)	(44209, 26)	(44210, 26)	RFC_dim_red w/o oversampling	0.28	0.23	0.20	0.49	0.65	285
(100, 1000)	(145065, 26)	(44210, 26)	SGD_dim_red w oversampling	0.00	0.01	0.00	0.01	0.01	285
(100, 1000)	(44209, 26)	(44210, 26)	SGD_dim_red w/o oversampling	0.00	0.01	0.00	0.01	0.01	285
(1000, 6000)	(37642, 186)	(22277, 186)	RFC_ohc w oversampling	0.49	0.47	0.44	0.76	0.85	29
(1000, 6000)	(22277, 186)	(22277, 186)	RFC_ohc w/o oversampling	0.49	0.47	0.44	0.76	0.84	29
(1000, 6000)	(37642, 186)	(22277, 186)	RFC_ohc w oversampling	0.50	0.45	0.41	0.73	0.83	29
(1000, 6000)	(22277, 186)	(22277, 186)	SGD_ohc w/o oversampling	0.52	0.46	0.41	0.73	0.84	29
(1000, 6000)	(37642, 26)	(22277, 26)	RFC_dim_red w oversampling	0.57	0.54	0.53	0.79	0.86	29
(1000, 6000)	(22277, 26)	(22277, 26)	RFC_dim_red w/o oversampling	0.56	0.54	0.53	0.79	0.86	29
(1000, 6000)	(37642, 26)	(22277, 26)	SGD_dim_red w oversampling	0.01	0.04	0.01	0.10	0.20	29
(1000, 6000)	(22277, 26)	(22277, 26)	SGD_dim_red w/o oversampling	0.02	0.05	0.02	0.05	0.10	29

Таблица 4. Результаты классификации на втором уровне исполнения. test_size=0.5.

Приложение F. Результаты классификации на третьем уровне исполнения

subset	train_size	test_size	method	precision	recall	f1	top_3	top_5	n_classes
(10, 100)	(36736, 651)	(12190, 651)	RFC_ohc w oversampling	0.28	0.25	0.23	0.51	0.63	656
(10, 100)	(12190, 651)	(12190, 651)	RFC_ohc w/o oversampling	0.27	0.25	0.23	0.52	0.64	656
(10, 100)	(36736, 651)	(12190, 651)	SGD_ohc w oversampling	0.26	0.25	0.22	0.52	0.67	656
(10, 100)	(12190, 651)	(12190, 651)	SGD_ohc w/o oversampling	0.25	0.25	0.21	0.52	0.66	656
(10, 100)	(36736, 29)	(12190, 29)	RFC_dim_red w oversampling	0.31	0.27	0.26	0.51	0.60	656
(10, 100)	(12190, 29)	(12190, 29)	RFC_dim_red w/o oversampling	0.30	0.28	0.27	0.53	0.62	656
(10, 100)	(36736, 29)	(12190, 29)	SGD_dim_red w oversampling	0.00	0.00	0.00	0.00	0.00	656
(10, 100)	(12190, 29)	(12190, 29)	SGD_dim_red w/o oversampling	0.00	0.00	0.00	0.00	0.00	656
(100, 500)	(40192, 422)	(15603, 422)	RFC_ohc w oversampling	0.47	0.44	0.42	0.76	0.84	157
(100, 500)	(15603, 422)	(15603, 422)	RFC_ohc w/o oversampling	0.47	0.46	0.44	0.77	0.86	157
(100, 500)	(40192, 422)	(15603, 422)	SGD_ohc w oversampling	0.48	0.45	0.41	0.78	0.89	157
(100, 500)	(15603, 422)	(15603, 422)	SGD_ohc w/o oversampling	0.50	0.46	0.42	0.77	0.88	157
(100, 500)	(40192, 29)	(15603, 29)	RFC_dim_red w oversampling	0.50	0.48	0.47	0.73	0.80	157
(100, 500)	(15603, 29)	(15603, 29)	RFC_dim_red w/o oversampling	0.49	0.48	0.47	0.75	0.82	157
(100, 500)	(40192, 29)	(15603, 29)	SGD_dim_red w oversampling	0.00	0.01	0.00	0.01	0.01	157
(100, 500)	(15603, 29)	(15603, 29)	SGD_dim_red w/o oversampling	0.00	0.02	0.00	0.02	0.03	157
(100, 1000)	(79344, 430)	(20842, 430)	RFC_ohc w oversampling	0.43	0.39	0.37	0.71	0.81	174
(100, 1000)	(20841, 430)	(20842, 430)	RFC_ohc w/o oversampling	0.43	0.40	0.38	0.72	0.82	174
(100, 1000)	(79344, 430)	(20842, 430)	SGD_ohc w oversampling	0.44	0.40	0.37	0.73	0.86	174
(100, 1000)	(20841, 430)	(20842, 430)	SGD_ohc w/o oversampling	0.46	0.42	0.39	0.73	0.85	174
(100, 1000)	(79344, 29)	(20842, 29)	RFC_dim_red w oversampling	0.44	0.41	0.41	0.69	0.77	174
(100, 1000)	(20841, 29)	(20842, 29)	RFC_dim_red w/o oversampling	0.45	0.42	0.41	0.72	0.80	174
(100, 1000)	(79344, 29)	(20842, 29)	SGD_dim_red w oversampling	0.00	0.00	0.00	0.01	0.01	174
(100, 1000)	(20841, 29)	(20842, 29)	SGD_dim_red w/o oversampling	0.00	0.00	0.00	0.01	0.02	174

Таблица 5. Результаты классификации на третьем уровне исполнения. test_size=0.5.