Job recommendation system report

Fabuary 2024

That project propose solution for job recommendations problem via topic modeling. Used combination topic model and cosine similarity for more correct job recommendations.

Contents

1 Introduction

Nowadays, students are constantly faced with the challenge of finding suitable employment opportunities that align with their skills and interests in modern competitive job market. The process of searching for job can often be overwhelming and time consuming, especially for students, because they often have no experience. In addition, students often look for work while studying at the university, so they cannot spend all time to job search due to academic studing. As such, effective and efficient job recommendation program is very useful for students. This program would help students both streamline their job search process and advice them job according their qualifications and career goals. Therefore, the development and implementation of jobs recommendation program is crucial in today's fast-paced and competitive job market. Nevertheless, this program can be useful not only for students, but also for those who have decided to change their career specialization. Moreover, it is important to know which skills will be very useful in the field of the profession for futher self-improvement. Also, one of the most unpleasant questions for students is the question of the approximate salary level. Most of the inexperienced applicants are very confuse by the salary question. Therefore, the proposed program offers not only job recommendations, but also recommendations on skills that are often used in the recommended field, as well as an approximate salary level.

2 Related work

In the work [1], the probability of an employee changing jobs is predicted depending on his skills, time spent working in the company and salary. This article discusses classical classification algorithms such as: LogisticRegression, SGDClassifier, DecisionTrees,NaiveBayes, RandomForest, AdaBoost, GradientBoosting, XGBoost, Cat Boost. The best result was shown by Cat Boost.

In [2], the problem of sequence tagging is considered. That is, in fact, based on the text, we classify it for an arbitrary number of classes. This article discusses models from the Bidirectional LSTM-CRF family with some author's modifications.

In [3], the use of neural networks -transformers is considered, and the classification takes place by majority vote. The authors considered TextCNN, Bi-GRU-CNN, Bi-GRU-CNN. Embeddings taken from Glove or FastText are received at the input of the models. The best result of the F1 score was obtained by an ensemble of the above models.

3 Dataset

One of the largest job search sites, hh.ru, was chosen as the data source. All data are publicly available. However, access to the data through anonymous queries has recently been made more difficult. Therefore, a relatively small number of vacancies with descriptions have been collected. The data collection resulted in

a table with 11,000 objects and 8 attributes: job index, company name, job title, experience, schedule, key skills, description, salary range. The table contains a lot of nan values, so before each stage, the data is cleaned of such values in the target attribute. in At the first stages, models were built on the basis of the entire text from job descriptions (with the exception of popular stop words in Russian), but after such preprocessing of the text, the models produced poor results. Most of the words in the keywords related to common words. They had nothing to do with specializations. Then there was an attempt to trim vacancies texts according to words from the middle (often there were such useful words as skills), but this led to the same result as before. Therefore, it was decided to create a list of all skills (taken from combining all values). The normalized words from the job description were filtered out so that only those from the specified list of all skills remained. However, there is such a problem that skills are written in two languages at the same time. For example, 'c++' with either English symbol 'c' or Russian symbol 'с' (the same case with 'c', '1c'). There are also cases when the skill is indicated together with a number or version ('c++17', 'kubernetes' and 'k8s', 'stm32', 'g4', '3gpp') or is generally indicated through a character ('c/c++'). In addition, there are also strange and too general skills that were immediately removed ('трубопровод', 'вахта', 'развитие', 'участие', 'активность' etc.). That problem was particular solved by filtering the list of skills and replacing skill tokens in the text, which mean the same thing (for example: 'k8s' → 'kubernetes', 'джава'→ 'java', 'js' → 'javascript', 'c++17' → 'c++').

4 Model description

4.1 Model concept

Program input is a short excerpt from the user's resume on skills. There are three main tasks:

- Selection of vacancies for the user
- Specifying skills that can be useful for a career
- Salary estimation

For a more correct job placement, it was decided to first divide all vacancies into professions and areas of specialization, and then select vacancies and in-demand skills in the selected area. After the selection of vacancies, a salary estimate will be issued. The division of all vacancies into areas of specialization is a topic modeling task. The process of selecting vacancies and skills is the task of building recommendations. And estimating salary based on a resume is a regression task.

4.2 Baseline solution

A similar task has been considered by one of the authors before. The task was given at a hackathon from the educational platform 'Netology'. It was also about helping beginners find a job. Three days were allotted to solve it. The author's team developed a model [1], but it was based on text cluster analysis and gave many inaccurate recommendations for not the most popular professions.

Unfortunately, the result of that model cannot be run for comparison because the data was taken from an anonymous request to the site. At the moment it is impossible to get such detailed data through an anonymous query.

Queries in the source are available as model results. The model worked correctly for web and frontend areas, android development, and programming in the most popular languages. But if we're introducing terms that are less popular, such as embedded, the model starts to to make a lot of inaccurate recommendations.

4.3 Topic modeling

4.3.1 Latent Dirichlet Allocation

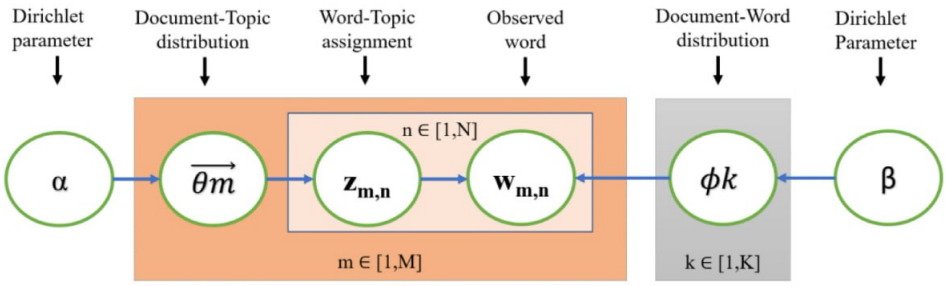
. The model was introduced as a graph model for detecting topics by David Bell, Andrew Son, and Michael Jordan in 2003 [4]. We prepare the data for training the model by composing bigrams and trigrams of words. After that, we turn them into embedding by composing a bag of words. We hypothesize that the frequency of words in various topics corresponds to a certain distribution law. The Dirichlet distribution is taken as this law (the general case of the Beta distribution law, for a multidimensional space) [you can insert the Dirichlet distribution formula]. First, the topic distributions for documents and word distributions for topics are initialized, then an iterative process with several operations is performed:

- For each word in each document, the probability of belonging to each topic is calculated using the current topic and word distributions.
- Based on the probabilities of the words in the topics and the probabilities of the topics in the documents, the distributions of topics and words are recalculated.

The Dirichlet distribution of order $K \geq 2$ with parameters $\alpha_1 \dots \alpha_k \geq 0$ has a probability density function

$$f(x_1 \dots x_K; \alpha_1 \dots \alpha_K) = \frac{1}{B(\alpha)} \prod_{i=1}^{K} x_i^{\alpha_i - 1}$$

$\sum_{i=1}^{K} x_i = 1$ and $x_i \in [0, 1]$ for all $\in \{1, \dots, K\}$ and $B(\alpha)$ is beta function



The model has many input hyperparameters. One of the most important parameters is the number of topics. Its selection was carried out through the training of models with different parameters and the selection of a model with the maximum coherence value of the model. Model Parameters:

- chunksize (int, optional) – Number of documents to be used in each training chunk.
- passes (int, optional) – Number of passes through the corpus during training.
- update_every (int, optional) – Number of documents to be iterated through for each update. Set to 0 for batch learning, > 1 for online iterative learning.
- alpha (float, numpy.ndarray of float, list of float, str, optional) – A-priori belief on document-topic distribution
- eta (float, numpy.ndarray of float, list of float, str, optional) – A-priori belief on topic-word distribution

4.3.2 Non-negative matrix factorization

Early work on non-negative matrix factorization was carried out by a Finnish group of researchers in the mid-1990s under the name positive matrix decomposition [2]. The algorithm itself belongs to the models of dimension reduction. When processing the text of vacancies, we will use the TF-IDF algorithm. Then TfidfVectorizer from the library sklearn was used to get a number for each word. The goal of using TF-IDF is to scale down the impact of tokens that occur very frequently in a given corpus and that are hence empirically less informative than features that occur in a small fraction of the training corpus. The formula describing TF-IDF embedding:

$$x_{ik} = f_{ik} log \frac{N}{n_i}$$

where i - word index, k — comment index, f_{ik}— word frequency in a comment, N — number of comments in the corpus, n_i — number of comments containing the word. Now there is a matrix V of dimension $m \times n$, where m is the number of words, n is the number of documents. NMF tries to find two matrices W and

H of dimensions $m \times k$ and $k \times n$, where k is the number of topics (factors) such that $V \approx WH$. The matrix W contains 'topic words' and shows which words characterize each topic. The H matrix contains 'document topics' and shows how much each topic is present in each document. NMF minimizes the difference between the original matrix V and the approximate matrix WH, taking into account the restriction on the non-negativity of all elements of the matrices W and H. This allows you to get interpretable results, since the elements of the matrices represent the importance of words and topics in the document collection. The approximation itself is achieved by minimizing the error functional $|(|V - WH|)|_F, W > 0, H > 0$.

4.3.3 Cluster analysis

This method involves applying clustering methods for thematic modeling. These methods are applied to the preprocessed feature matrix from the original document set. Only the first and second approaches were implemented. The third method was used by one of the authors to solve a similar problem three years ago. The comparison of model recommendations will be presented in the 'Results' section.

4.4 Recommendation system

Cosine similarity formula:

$$\cos(\theta) = \frac{A \cdot B}{||A||||B||}$$

The method of collaborative filtering based on cosine distance was chosen as a method of selecting recommendations. For the processed resume, we consider the matrix of processed vacancy texts, calculate the cosine distance for each vacancy, then find the calculated vectors with the highest norm. Then we calculate the same thing for each skill instead of vacancies, and again find the vectors with the largest cosine distance. As a result, we get a set of skills and vacancies for the applicant's resume submitted to the input model. It is worth noting that the labels of the areas of specialization can be obtained using both the NMF algorithm and LDA.

4.5 Salary estimation

It is worth noting right away that only an approximate estimate is needed, which the user can navigate so as not to get lost, so there is not much point in using very heavy models. Before model building some object was deleted, because it salary feature look strange and too high. There are a lot of different implementations of regression models. In our work, the CatBoostRegressor model is used, since it works quickly, is optimized for categorical variables, has good accuracy with basic hyperparameters, is stable with respect to their changes, has support for learning on video cards, and can accept text signs as input. The model has a lot of parameters, but only the following were used:

- text_features – text features in the data
- cat_features – categorical features
- loss_function – optimized loss function
- learning_rate – learning rate parameter
- iterations – the number of iterations in the algorithm
- depth – the depth of the decision tree
- verbose – parameter for displaying model metrics
- random _state – parameter of the random number generator

There are features in the data collected that directly affect salary. It is schedule and also job experience. They represent categorical data that are a perfect fit for the specified model. After training, a file is saved in which for each variant with a schedule and job experience salary estimate is given for the submitted resume. This can be useful in planing the prospects and salary growth with incresing experience and extend job schedule.

5 Train process

5.1 Metrics

Topic modeling is quite a specific task for NLP. It doesn't require splitting the data into training and validation sets, and it's also difficult to evaluate the quality of such models using any metrics. Therefore, the analysis is often limited to reviewing keywords for different topics. In this study, the approximate quality assessment for the LDA model is based on perplexity and model coherence. For the NMF model, the cosine similarity matrix between different topic representations and the topic/keyword matrix are generated. In addition to the mentioned metrics, during the model building process, a file is saved where keywords for different topics are stored so that the user can visually evaluate the model.

Model Perplexity formula:

$$perplexity(D) = \exp(-\frac{\sum \log p(w)}{\sum_{d=1}^{M} N_d})$$

Model Coherence formula:

$$v(W') = \sum_{w_i \in W'} NPMI(w_i, w_j)^{\gamma}_{j=1,\cdots|W|}$$

$$NPMI(w_i, w_j)^{\gamma} = (\frac{\log \frac{P(w_i,w_j)+\epsilon}{P(w_i)\cdot P(w_j)}}{-\log(P(w_i, w_j) + \epsilon)})$$

$$\phi_{S_i}(\overrightarrow{u}, \overrightarrow{w}) = \frac{\sum_{i=1}^{W} u_i \cdot w_i}{||u||_2 \cdot ||w||_2}$$

There are coherence score description based on empirical experiments in topic modeling:

- 0.3 is bad
- 0.4 is low
- 0.55 is okay
- 0.65 might be as good as it is going to get
- 0.7 is nice
- 0.8 is unlikely
- 0.9 is probably wrong

5.2 Models configuration

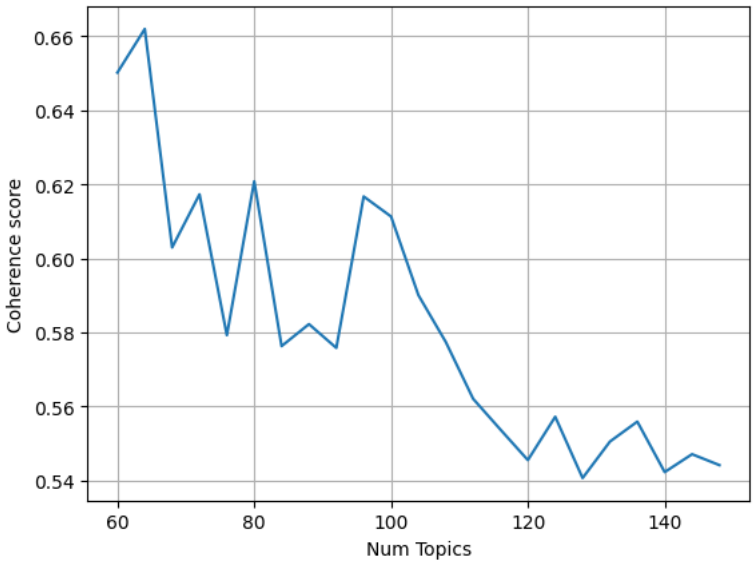
LDA model Input preprocess:

- Creation words bigrams and trigrams
- Creation bag-of-word

Encoding words model hyperparameters:

- num _topics – 64,
- eta: 0.8
- alpha: “auto”
- random _state:
- update _every: 1
- chunksize: 100
- passes: 5

The most important parameter in LDA is undoubtedly the number of topics. Changing it can lead to completely different model results. The selection of this parameter was done by iterating through models and plotting their coherence scores on a graph. The highest coherence score (up to 0.8) would be the best for the model.



NMF model Input preprocess: TF-IDF encoding:

- max _features=1000
- max _df=0.997,
- min _df=0.003

Model hyperparameters:

- n _components: 120
- random _state: 0,
- solver: 'mu',
- beta _loss: float or {'frobenius', 'kullback-leibler', 'itakura-saito'}, default='frobenius' - Beta divergence to be minimized, measuring the distance between X and the dot product WH. Where d_{KL} - kullback-leibler distance between X and the dot product WH.

$$d_{KL}(X,Y) = \sum_{i,j}(X_{ij}\log(\frac{X_{ij}}{Y_{ij}}) - X_{ij} + Y_{ij})$$

In the NMF model, the parameter for the number of topics (n _components) is not as sensitive to changes, so the number is chosen after several program runs and visual evaluation of keywords in the topics.

Some words about regression model. Training the CatBoostRegressor model is typical task for machine learning. The model itself is relatively stable when it comes to changing parameters. The square root of the standard quadratic

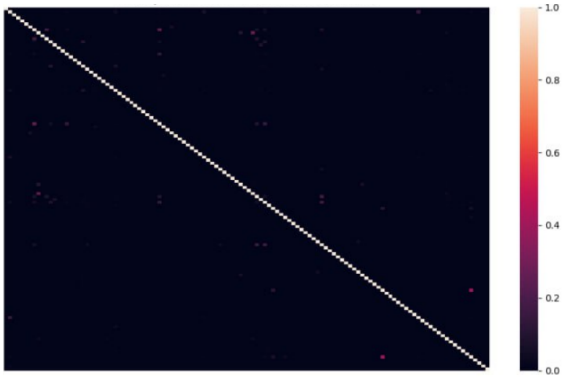


Рис. 1: Cosine similirity between all topics

loss function was chosen as the loss function. To translate it into an absolute scale, this indicator was normalized by the difference between the maximum and minimum values, as the loss function depends on the magnitude of numbers in the training sample. During training without any preprocessing, a loss of 0.2 was achieved. However, outlies removing (salary greater than 500000 or less than 40000), the loss improved to 0.08.

Model hyperparameters:

- text _features: ['Description']
- cat_features: ['Experience', 'Schedule']
- loss _function: 'RMSE'
- learning _rate: 0.25
- iterations: 200
- depth: 7
- verbose: 30
- random _state: 0

6 Results

The most illustrative results for the task at hand can only be obtained through visual assessment of the program execution. Therefore, let's create several descriptions of candidates' skills and input them into the model. It should be noted that the program may not work correctly if the applicant tries to find a job with skills that are rarely encountered in the dataset or describes their skills incorrectly.

Plus, the program will display additional information about the recommended profession and the most popular skills in that profession as it runs.

We make three queries to check the final accuracy of the model. The first query will consist of popular words in the dataset. The second query will contain less popular words. And the third query will consist of rare words.

number	query
1	“python, nlp, algorithm, нейросети”
2	”микроконтроллер, stm32, c”
3	”design, photoshop, figma, 3d, ux-ui”

skill	similarity
opengl	0.02679
рефакторинг	0.03072
cuda	0.03907
api	0.12074
библиотека	0.18294
алгоритм	0.23379
ml	0.32840

Таблица 1: LDA skill recommendations 1 query

skill	similarity
sota	0.10307
transformers	0.14428
gpt	0.15815
dl	0.22183
llm	0.17912
bert	0.17508
pytorch	0.29088

Таблица 2: NMF skill recommendations 1 query

.

.

	Name	Keys
0	Педагог дополнительного о...	"['Обучение и развитие', 'Java', 'CM..."
1	HR manager / Рекрутер (IT...	"['Подбор персонала', 'Подбор сотруд..."
2	Junior Data Engineer	"['Hive', 'Hadoop', 'Аналитическое м..."
3	"Специалист по ML, Data Sc..."	"['ML', 'AI', 'GPT', 'Tensorflow', '..."
4	"Преподаватель курсов Про..."	"['Python', 'NumPy', 'Matplotlib', '..."

Таблица 3: LDA job recommendations 1 query

	Name	Keys
0	ML инженер	['ML']
1	Data Scientist (Middle/Se...	"['ML', 'NLP', 'PyTorch', 'BERT', 'S..."
2	Руководитель направления ...	
3	Middle+ NLP Engineer	"['Python', 'Git', 'Linux', 'Docker'..."
4	Data scientist (команда C...	"['SQL', 'Python', 'Математическое м..."

Таблица 4: NMF job recommendations 1 query

	Schedule	Experience	Salary
0	Вахтовый метод	Более 6 лет	221634.6518035233
1	Вахтовый метод	Нет опыта	124804.92014458776
2	Вахтовый метод	От 1 года до 3 лет	135324.89975884557
3	Вахтовый метод	От 3 до 6 лет	194473.42417290807
4	Гибкий график	Более 6 лет	191955.14479592443
5	Гибкий график	Нет опыта	81922.68747094274
6	Гибкий график	От 1 года до 3 лет	92442.66708520055
7	Гибкий график	От 3 до 6 лет	156208.92381814122
8	Полный день	Более 6 лет	187584.23854592443
9	Полный день	Нет опыта	83798.70512154698
10	Полный день	От 1 года до 3 лет	94318.6847358048
11	Полный день	От 3 до 6 лет	151838.01756814122
12	Сменный график	Нет опыта	83798.70512154698
13	Сменный график	От 1 года до 3 лет	94318.6847358048
14	Сменный график	От 3 до 6 лет	151838.01756814122
15	Удаленная работа	Более 6 лет	201271.20045998693
16	Удаленная работа	Нет опыта	87010.95138695836
17	Удаленная работа	От 1 года до 3 лет	97530.93100121617
18	Удаленная работа	От 3 до 6 лет	161297.18773415685

Таблица 5: Salary estimation 1 query

.

skill	similarity
алгоритм	0.04082
архитектура	0.04385
тестирование	0.05257
радиотехник	0.11963
мультиметр	0.10515
Ethernet	0.27190
c	0.00157

Таблица 6: LDA skills recommendations 2 query

skill	similarity
git	0,06474
embedded	0,15956
arm	0,24307
Ethernet	0,29826
схемотехник	0,41116
uart	0,45600
i2c	0,42967

Таблица 7: NMF skill recommendations 2 query

	Name	Keys
0	Специалист службы поддерж...	
1	Главный инспектор отдела ...	
2	Тестировщик ПО	"['Atlassian Jira', 'Функциональное ..."
3	Региональный менеджер по ...	"['Ведение переговоров', 'Развитие п..."
4	Главный инженер по информ...	

Таблица 8: LDA job recommendations 2 query

	Name	Keys
0	Инженер-программист микро...	"['C/C++', 'Разработка ПО', 'STM32',..."
1	Программист-робототехник	
2	Инженер-программист (Embe...	"['Английский язык', 'Keil', 'Микрок..."
3	Программист встроенных си...	"['C/C++', 'Linux', 'ARM', 'STM32', ..."
4	Инженер-программист микро...	['Техническая поддержка разрабатыва...

Таблица 9: NMF job recommendations 2 query

	Schedule	Experience	Salary
0	Вахтовый метод	Более 6 лет	200188.80112668872
1	Вахтовый метод	Нет опыта	135227.84936735034
2	Вахтовый метод	От 1 года до 3 лет	141796.8540136516
3	Вахтовый метод	От 3 до 6 лет	173027.57349607348
4	Гибкий график	Более 6 лет	170509.29411908984
5	Гибкий график	Нет опыта	92345.61669370532
6	Гибкий график	От 1 года до 3 лет	98914.62134000659
7	Гибкий график	От 3 до 6 лет	134763.07314130664
8	Полный день	Более 6 лет	158673.67040082812
9	Полный день	Нет опыта	83843.70360776782
10	Полный день	От 1 года до 3 лет	90412.70825406909
11	Полный день	От 3 до 6 лет	122927.44942304492
12	Сменный график	Нет опыта	83843.70360776782
13	Сменный график	От 1 года до 3 лет	90412.70825406909
14	Сменный график	От 3 до 6 лет	122927.44942304492
15	Удаленная работа	Более 6 лет	179825.34978315234
16	Удаленная работа	Нет опыта	97433.88060972095
17	Удаленная работа	От 1 года до 3 лет	104002.88525602221
18	Удаленная работа	От 3 до 6 лет	139851.33705732226

Таблица 10: Salary estimation 2 query

skill	similarity
low	0.03889
cross	0.04761
science	0.05474
system	0.07750
development	0.09226

Таблица 11: LDA skill recommendations 3 query

skill	similarity
яндекс	0,03159
микросервисы	0,03321
load	0,03555
managment	0,05640
low	0,03889

Таблица 12: NMF skill recommendations 3 query

	Name	Keys
0	Системный аналитик в стар...	"['SQL', 'Аналитика', 'Разработка те..."
1	Data Analyst	"['Python', 'SQL', 'Анализ данных', ..."
2	UX/UI проектировщик интер...	['Проектирование пользовательских и...
3	Ведущий инженер службы по...	"['Пользователь ПК', 'Контроль качес..."
4	Главный инженер	

Таблица 13: LDA job recommendations 3 query

	Name	Keys
0	UI/UX designer	"['Figma', 'Adobe Photoshop', 'Англи..."
1	Senior Frontend-разработч...	['JavaScript']
2	UX/UI дизайнер	"['UX', 'UI', 'Дизайн интерфейсов', ..."
3	Junior Designer	"['Графический дизайн', 'Дизайн инте..."
4	UX/UI проектировщик интер...	['Проектирование пользовательских и...

Таблица 14: NMF job recommendations 3 query

	Schedule	Experience	Salary
0	Вахтовый метод	Более 6 лет	173160.0526356399
1	Вахтовый метод	Нет опыта	108199.10087630153
2	Вахтовый метод	От 1 года до 3 лет	114768.1055226028
3	Вахтовый метод	От 3 до 6 лет	145998.82500502467
4	Гибкий график	Более 6 лет	143480.54562804103
5	Гибкий график	Нет опыта	65316.86820265651
6	Гибкий график	От 1 года до 3 лет	71885.87284895778
7	Гибкий график	От 3 до 6 лет	107734.32465025783
8	Полный день	Более 6 лет	131644.9219097793
9	Полный день	Нет опыта	56814.95511671901
10	Полный день	От 1 года до 3 лет	63383.95976302028
11	Полный день	От 3 до 6 лет	95898.7009319961
12	Сменный график	Нет опыта	56814.95511671901
13	Сменный график	От 1 года до 3 лет	63383.95976302028
14	Сменный график	От 3 до 6 лет	95898.7009319961
15	Удаленная работа	Более 6 лет	152796.60129210353
16	Удаленная работа	Нет опыта	70405.13211867213
17	Удаленная работа	От 1 года до 3 лет	76974.1367649734
18	Удаленная работа	От 3 до 6 лет	112822.58856627345

Таблица 15: Salary estimation 3 query

One can notes for first query, that NMF return more specific skills for nlp, at the same time LDA model return more regular skills for all artificial intelligence field. However, both recomendations are good. But the difference in LDA and NMF

recommendations becomes significant for the second query. Obviously, NMF return better recomendations for jobs. Although LDA recommendations are still not bad for skills. After third query both model return bad skills recommendations. Those recomendations have nothing to do with query skills. But NMF jobs recommendations are still pretty good, unlike the LDA model, who's not doing so well either.

7 Conclusion

The NMF model showed the best result. It provides recommendations significantly better than LDA, especially for rare specializations. However, all models give inaccurate recommendations for skills in the third query. The baseline solution is inferior to both NMF and LDA. Additionally, NMF has significantly fewer parameters to tune. Moreover, it is less sensitive to changes in these parameters. It can also be seen that for the third query, NMF produces good recommendations as opposed to skills. The disadvantage is that skills are processed on a word-by-word basis, so sometimes only one part of a word combination is visible in the skills recommendations (e.g. "библиотека"). Based on these results, one can be said that for this recommendation method, the NMF model is the best topic model, showing stable and accurate recommendations.

References

1. Dr.Padmaja Pulicherla ,Teegala Kirshna Reddy: Job Shifting Prediction and Analysis Using Machine Learning, 2019

2. Zhiheng Huang, Wei Xu, Kai Yu: Bidirectional LSTM-CRF Models for Sequence Tagging, 2015

3. Tin Van Huynh, Kiet Van Nguyen, Ngan Luu-Thuy Nguyen and Anh Gia-Tuan Nguyen: Job Prediction - From Deep Neural Network Models to Applications, 2020

4. Blei, David M. Ng, Andrew Y., Jordan, Michael I.: Latent Dirichlet allocation , 2003.

5. Paatero P., Tapper U.: Positive matrix factorization - A non-negative factor model with optimal utilization of error estimates of data values // Environmetrics. — 1994

6. Anna Veronika Dorogush, Vasily Ershov, Andrey Gulin: CatBoost - gradient boosting with categorical features support, 2017