Министерство науки и высшего образования Российской Федерации
Федеральное государственное автономное образовательное учреждение высшего образования
«Московский государственный технический университет имени Н.Э. Баумана
(национальный исследовательский университет)»
(МГТУ им. Н.Э. Баумана)

Факультет «Информатика и системы управления»

Кафедра «Системы обработки информации и управления»

Курс «Технологии машинного обучения»
Отчёт по лабораторной работе №1

Выполнила:
студентка группы ИУ5-62Б Вешторт Е.С.
Подпись:
Проверил:
Гапанюк Ю.Е.
Подпись:

2025 г.

1. Определение данных для анализа

В качестве данных для анализа был выбран датасет «LinkedIn Job Postings (2023 - 2024)».

В датасете представлены все предложения о работе, выложенные на сайте LinkedIn с конца марта 2024 года по конец апреля.

2. Описание данных

Для анализа были выбраны три таблицы датасета: postings (информация о размещенных объявлениях), benefits (описание дополнительных преимуществ вакансий) и employee_counts (информация о количестве сотрудников в компаниях)

postings:

- job_id — ID вакансии
- company_name — Название компании
- title — Название должности
- description — Описание
- max_salary — Максимальная зарплата
- pay_period — Период выплаты
- location — Местоположение
- company_id — ID компании
- views — Количество просмотров
- med_salary — Медианная зарплата
- min_salary — Минимальная зарплата
- formatted_work_type — Формат работы (например, удалённая, офисная)
- applies — Количество откликов
- original_listed_time — Время изначальной публикации
- remote_allowed — Разрешена удалённая работа
- job_posting_url — URL публикации вакансии
- application_url — URL для подачи заявки

- application_type — Тип подачи заявки
- expiry — Дата истечения срока действия вакансии
- closed_time — Дата закрытия вакансии
- formatted_experience_level — Уровень опыта
- skills_desc — Описание навыков
- listed_time — Дата публикации вакансии
- posting_domain — Домен публикации
- sponsored — Спонсируемая вакансия (0 или 1)
- work_type — Тип работы
- currency — Валюта
- compensation_type — Тип компенсации
- normalized_salary — Нормализованная зарплата
- zip_code — Почтовый индекс
- fips — Код федеральной информационной обработки (FIPS)

benefits:

- job_id — ID вакансии
- inferred — Выведено (или Предполагаемое значение)
- type — Тип вознаграждения

employee_counts:

- company_id — ID компании
- employee_count — Количество сотрудников
- follower_count — Количество подписчиков
- time_recorded — Время записи

3. Формулирование гипотез

Гипотеза 1: У вакансий с полной занятостью в среднем больше дополнительных вознаграждений (пенсия, страховка), чем у вакансий с частичной занятостью.

Гипотеза 2: При наличие дополнительных вознаграждений (страховки, пенсии) средняя зарплата ниже при том же виде трудоустройства.

Гипотеза 3: Чем длиннее описание вакансии, тем выше конверсия из просмотра в отклик.

Гипотеза 4: Объявления с указанной информацией о зарплате чаще бывают успешными (успешными считаем закрытые объявления, неуспешными – объявления с истекшим сроком размещения).

Гипотеза 5: В среднем, в компаниях, которые разрешают удаленную работу, больше сотрудников, чем в тех, которые не разрешают

4. Подготовка данных для работы

Загружаем датасет и подключаем необходимые библиотеки

Выбираем из датасета поля, необходимые для анализа

```
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
```

```
pd.set_option('display.max_columns', None)
```

```
postings = pd.read_csv("dataset/postings.csv")
employee_counts = pd.read_csv("dataset/companies/employee_counts.csv")
benefits = pd.read_csv("dataset/jobs/benefits.csv")
```

```
postings = postings[['job_id', 'description', 'max_salary', 'min_salary',
'company_id', 'work_type', 'views', 'applies', 'pay_period', 'currency', 'listed_time', 'expiry', 'remote_allowed', 'closed_time']]
employee_counts = employee_counts.groupby('company_id')['employee_count'].mean().reset_index(name='employee_count')
benefits = benefits.groupby('job_id').size().reset_index(name='count_benefits')
```

Заполняем пропуски, отбрасываем строки без id компании:

```
postings['description'].fillna("", inplace=True)
postings['views'].fillna(0, inplace=True)
postings['applies'].fillna(0, inplace=True)
postings['remote_allowed'] = postings['remote_allowed'].apply(lambda x: True if x == 1 else False)
postings = postings.dropna(subset=['company_id'])
```

Преобразуем типы данных, объединяем три датафрейма в один, заполняем пропуски:

```
postings['listed_time'] = pd.to_datetime(postings['listed_time'], unit = 'ms')
postings['expiry'] = pd.to_datetime(postings['expiry'], unit = 'ms')
postings['closed_time'] = pd.to_datetime(postings['closed_time'], unit = 'ms')
postings['company_id'] = postings['company_id'].astype('int64')
```

```
postings = pd.merge(postings, employee_counts, on='company_id', how='left')
postings = pd.merge(postings, benefits, on='job_id', how='left')
```

```
postings['employee_count'].fillna(0, inplace=True)
postings['count_benefits'].fillna(0, inplace=True)
```

В нашем датасете данные о зарплате в разрозненном виде: где-то это зарплата за час, где-то за месяц, также зарплата указана в разной валюте. Для удобства анализа вычислим поле “зарплата за час в долларах” опираясь на количество рабочих часов в указанном периоде оплаты, а также на курс валют по отношению к доллару

```
currency_to_usd = {
    'USD': 1,
    'CAD': 0.75,  # Примерный курс
    'BBD': 0.5,   # Примерный курс
    'EUR': 1.1,   # Примерный курс
    'GBP': 1.3,   # Примерный курс
}

# Часы в рабочем периоде
pay_period_to_hours = {
    'HOURLY': 1,  # В час
    'YEARLY': 2080,  # Среднее количество рабочих часов в году
    'MONTHLY': 160,  # Среднее количество рабочих часов в месяц
    'WEEKLY': 40,   # Количество рабочих часов в неделю
    'BIWEEKLY': 80,  # Количество рабочих часов за две недели
}

# Функция для вычисления средней зарплаты в долларах за час
def calculate_hourly_rate(row):
    # Средняя зарплата в исходной валюте
    avg_salary = (row['min_salary'] + row['max_salary']) / 2

    # Приводим зарплату к долларам (с учетом курса валют)
    avg_salary_usd = avg_salary * currency_to_usd.get(row['currency'], 1)

    # Получаем количество рабочих часов в соответствующем pay_period
    hours_in_period = pay_period_to_hours.get(row['pay_period'], 40)  # если значение неизвестно, по умолчанию считаем 40 часов

    # Рассчитываем зарплату за час
    hourly_rate = avg_salary_usd / hours_in_period

    return hourly_rate
```

```
postings['hourly_rate_usd'] = postings.apply(calculate_hourly_rate, axis=1)
```

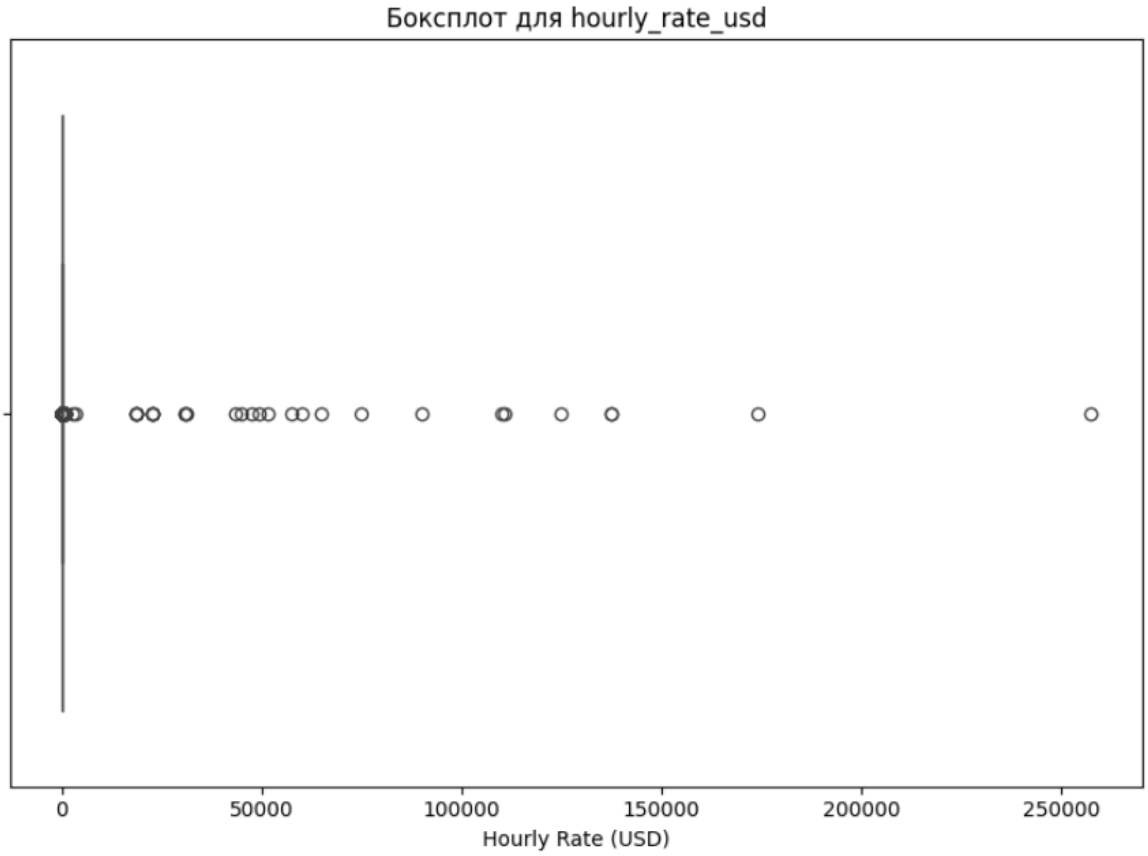
Добавим другие вычислемые поля, которые пригодятся при анализе:

```
postings['hourly_rate_usd'] = postings.apply(calculate_hourly_rate, axis=1)
```

```
postings['salary_stated'] = postings['hourly_rate_usd'].notna()
postings['description_length'] = postings['description'].str.len()
postings['listed_date'] = postings['listed_time'].dt.date
postings['benefits_group'] = postings['count_benefits'].apply(lambda x: 'With Benefits' if x > 0 else 'Without Benefits')
postings['apply_conversion'] = postings['applies'] / postings['views']
postings.loc[postings['views'] == 0, 'apply_conversion'] = 0
postings['hourly_rate_available'] = postings['hourly_rate_usd'].notnull()
```

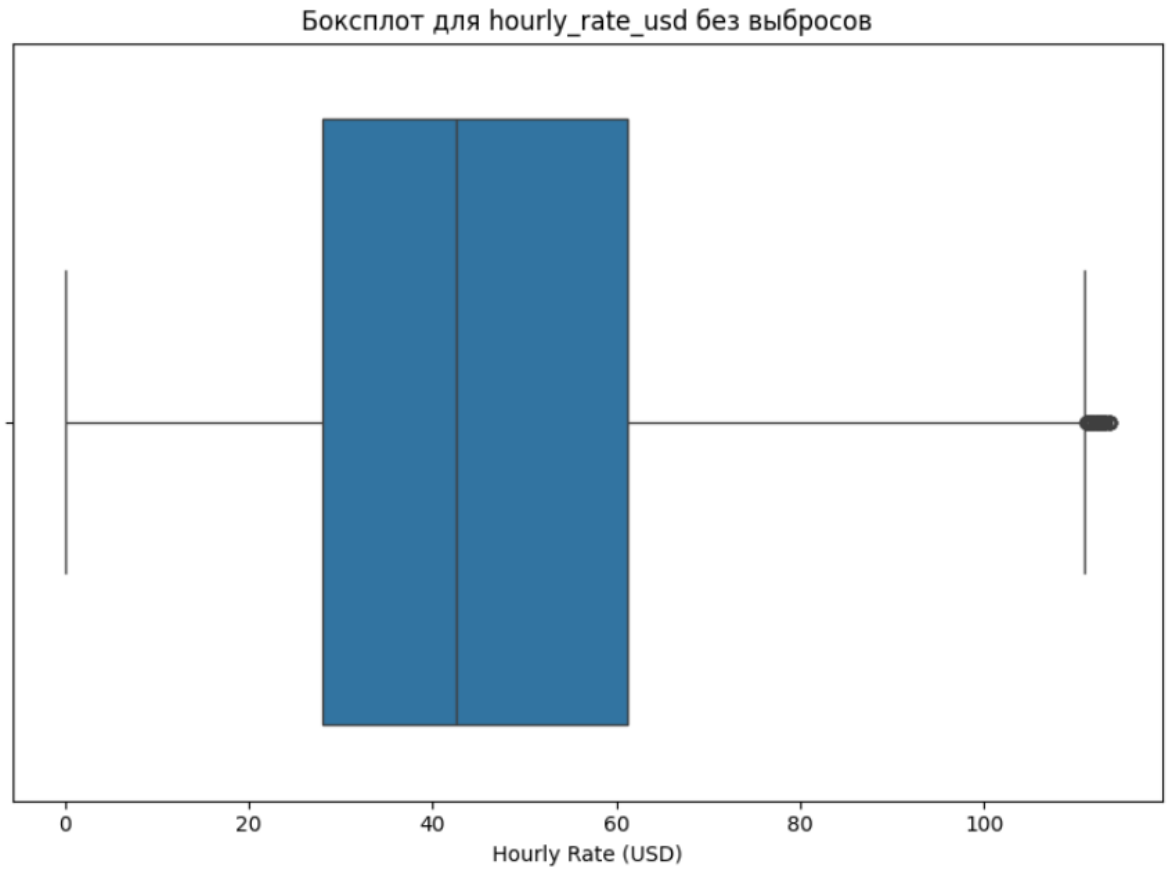
Построим боксплот для определения наличия выбросов:

```
# Строим боксплот
plt.figure(figsize=(8, 6))
sns.boxplot(data=postings, x='hourly_rate_usd')

# Настройки графика
plt.title('Боксплот для hourly_rate_usd')
plt.xlabel('Hourly Rate (USD)')

# Показать график
plt.tight_layout()
plt.show()
```



Видим, что в датасете есть выбросы по зарплате, уберем их:

```
Q1 = postings['hourly_rate_usd'].quantile(0.25)  # Первый квартиль (25%)
Q3 = postings['hourly_rate_usd'].quantile(0.75)  # Третий квартиль (75%)
IQR = Q3 - Q1  # Межквартильный размах

# Определяем границы для выбросов
lower_bound = Q1 - 1.5 * IQR  # Нижняя граница
upper_bound = Q3 + 1.5 * IQR  # Верхняя граница

# Фильтруем данные, исключив выбросы
postings = postings[(postings['hourly_rate_usd'].isnull()) | ((postings['hourly_rate_usd'] >= lower_bound) & (postings['hourly_rate_usd'] <= upper_bound))]

# Строим боксплот без выбросов
plt.figure(figsize=(8, 6))
sns.boxplot(data=filtered_data, x='hourly_rate_usd')

# Настройки графика
plt.title('Боксплот для hourly_rate_usd без выбросов')
plt.xlabel('Hourly Rate (USD)')

# Показать график
plt.tight_layout()
plt.show()
```



После произведенных манипуляций датасет готов для исследования

5. Изучение общей информации

```
<class 'pandas.core.frame.DataFrame'>
Index: 121282 entries, 0 to 122131
Data columns (total 23 columns):
 #   Column                 Non-Null Count   Dtype
---  ------                 --------------   -----
 0   job_id                 121282 non-null  int64
 1   description            121282 non-null  object
 2   max_salary             28489 non-null   float64
 3   min_salary             28489 non-null   float64
 4   company_id             121282 non-null  int64
 5   work_type              121282 non-null  object
 6   views                  121282 non-null  float64
 7   applies                121282 non-null  float64
 8   pay_period             34714 non-null   object
 9   currency               34714 non-null   object
 10  listed_time            121282 non-null  datetime64[ns]
 11  expiry                 121282 non-null  datetime64[ns]
 12  remote_allowed         121282 non-null  bool
 13  closed_time            1052 non-null    datetime64[ns]
 14  employee_count         121282 non-null  float64
 15  count_benefits         121282 non-null  float64
 16  hourly_rate_usd        28489 non-null   float64
 17  salary_stated          121282 non-null  bool
 18  description_length     121282 non-null  int64
 19  listed_date            121282 non-null  object
 20  benefits_group         121282 non-null  object
 21  apply_conversion       121282 non-null  float64
 22  hourly_rate_available  121282 non-null  bool
dtypes: bool(3), datetime64[ns](3), float64(8), int64(3), object(6)
memory usage: 19.8+ MB
```

Статистика по числовым столбцам:

	job_id	max_salary	min_salary	company_id	views
count	1.221320e+05	2.933900e+04	2.933900e+04	1.221320e+05	122132.000000
mean	3.897047e+09	9.225076e+04	6.511227e+04	1.220401e+07	14.436528
min	9.217160e+05	1.000000e+00	1.000000e+00	1.009000e+03	0.000000
25%	3.894609e+09	4.868000e+01	3.750000e+01	1.435200e+04	3.000000
50%	3.902312e+09	8.000000e+04	6.020000e+04	2.269650e+05	4.000000
75%	3.904709e+09	1.400000e+05	1.000000e+05	8.047188e+06	7.000000
max	3.906267e+09	1.200000e+08	8.500000e+07	1.034730e+08	9975.000000
std	7.138644e+07	7.064346e+05	4.997470e+05	2.554143e+07	85.676490

	applies	listed_time
count	122132.000000	122132
mean	1.975215	2024-04-15 18:18:26.233190400
min	0.000000	2024-03-24 21:50:14
25%	0.000000	2024-04-12 01:49:24
50%	0.000000	2024-04-18 02:35:30
75%	0.000000	2024-04-18 23:45:28
max	967.000000	2024-04-20 00:26:56
std	13.149750	NaN

	expiry	closed_time
count	122132	1056
mean	2024-05-20 03:07:12.453943552	2024-04-12 12:16:37.588068096
min	2024-04-12 06:30:48	2024-04-05 19:38:52
25%	2024-05-12 02:24:52	2024-04-09 13:36:00
50%	2024-05-18 14:21:55	2024-04-09 13:38:26
75%	2024-05-19 03:01:12	2024-04-16 15:45:33.249999872
max	2024-10-17 00:26:36	2024-04-19 21:28:27
std	NaN	NaN

	employee_count	count_benefits	hourly_rate_usd	description_length
count	122132.000000	122132.000000	29339.000000	122132.000000
mean	17484.367574	0.530590	114.107457	3793.860258
min	0.000000	0.000000	0.000481	0.000000
25%	187.500000	0.000000	28.725962	2209.000000
50%	1684.500000	0.000000	43.269231	3464.000000
75%	11245.000000	0.000000	62.740385	5012.000000
max	748029.500000	12.000000	257500.000000	23201.000000
std	59235.373675	1.296706	2717.271262	2141.012264

Количество уникальных значений в категориальных столбцах:

description	106140
work_type	7
pay_period	5
currency	5

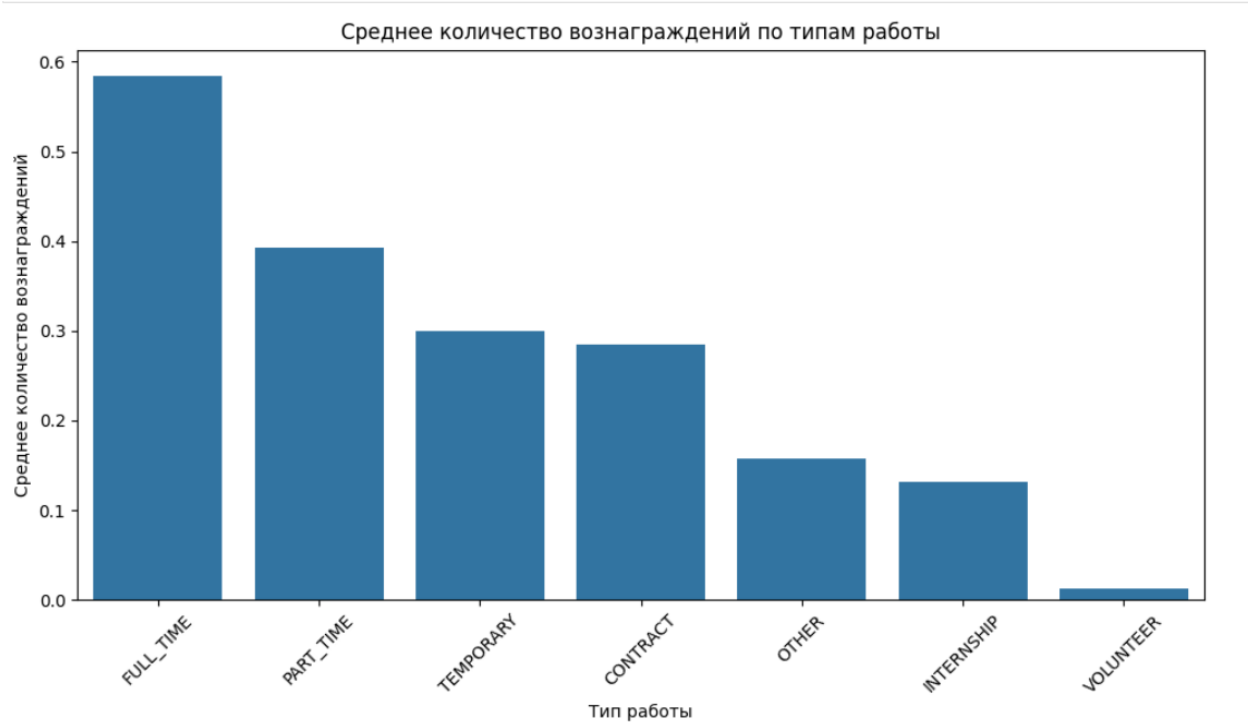
6. Исследовательский анализ данных

Гипотеза 1 - У вакансий с полной занятостью в среднем больше дополнительных вознаграждений (пенсия, страховка), чем у вакансий с частичной занятостью

Для проверки этой гипотезы построим график среднего количества дополнительных вознаграждений в разрезе типа занятости:

```
# Группируем данные по 'work_type' и вычисляем среднее значение для 'count_benefits'
average_benefits_by_work_type = postings.groupby('work_type')['count_benefits'].mean().reset_index()

# Сортируем данные по 'count_benefits' в убывающем порядке
average_benefits_by_work_type = average_benefits_by_work_type.sort_values(by='count_benefits', ascending=False)

# Строим столбчатую диаграмму с использованием seaborn
plt.figure(figsize=(10, 6))
sns.barplot(data=average_benefits_by_work_type, x='work_type', y='count_benefits')

# Настройки графика
plt.title('Среднее количество вознаграждений по типам работы')
plt.xlabel('Тип работы')
plt.ylabel('Среднее количество вознаграждений')
plt.xticks(rotation=45)  # Поворот меток на оси X для лучшей читаемости

# Показать график
plt.tight_layout()
plt.show()
```

Полученный график выглядит так:



Вывод:

На графике видим, что наибольшее количество вознаграждений, в среднем, имеют сотрудники с полным типом занятости, меньше – с частичной и временной занятостью. Меньше всего вознаграждений имеют волонтеры.

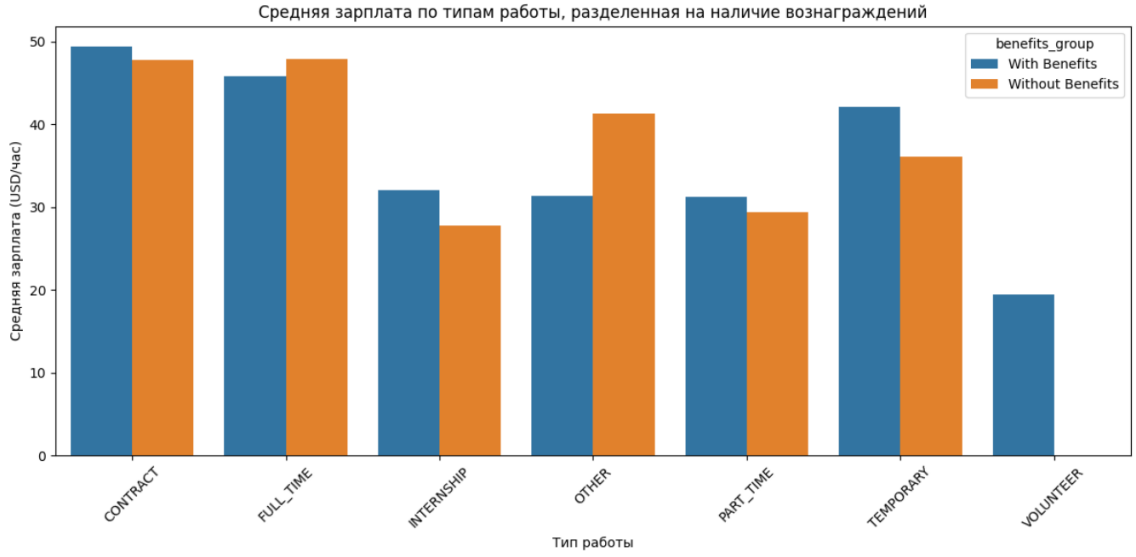
Гипотеза 1 – подтверждена

Гипотеза 2 - При наличие дополнительных вознаграждений (страховки, пенсии) средняя зарплата ниже при том же виде трудоустройства

Для проверки этой гипотезы построим график средней почасовой оплаты в долларах в зависимости от типа занятости и наличия/отсутствия дополнительных вознаграждений:

```
# Группируем данные по 'work_type' и 'benefits_group', и вычисляем среднее значение для 'hourly_rate_usd'
average_salary_by_benefits = postings.groupby(['work_type', 'benefits_group'])['hourly_rate_usd'].mean().reset_index()

# Строим столбчатую диаграмму с использованием seaborn
plt.figure(figsize=(12, 6))
sns.barplot(data=average_salary_by_benefits, x='work_type', y='hourly_rate_usd', hue='benefits_group')

# Настройки графика
plt.title('Средняя зарплата по типам работы, разделенная на наличие вознаграждений')
plt.xlabel('Тип работы')
plt.ylabel('Средняя зарплата (USD/час)')
plt.xticks(rotation=45)  # Поворот меток на оси X для лучшей читаемости

# Показать график
plt.tight_layout()
plt.show()
```

Полученный график выглядит так



Вывод:

На графике видим незначительную разницу между средней зарплатой работников с дополнительными вознаграждениями и без, которая, к тому же, разнонаправлена для различных видов трудоустройства.

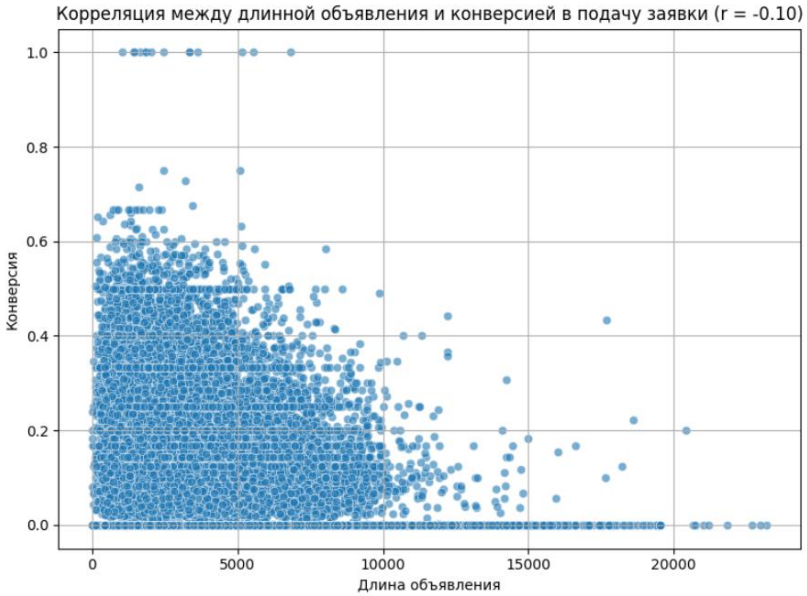
Гипотеза 2 не подтвердилась.

Гипотеза 3 - Чем длиннее описание вакансии, тем выше конверсия из просмотра в отклик

Для проверки этой гипотезы построим график и посчитаем коэффициент корреляции между длиной описания вакансии и конверсией из просмотра в отклик

```
# Рассчитываем коэффициент корреляции
correlation = postings['description_length'].corr(postings['apply_conversion'])

# Печатаем коэффициент корреляции
print(f"Коэффициент корреляции: {correlation:.2f}")

# Строим точечный график
plt.figure(figsize=(8, 6))
sns.scatterplot(data=postings, x='description_length', y='apply_conversion', alpha=0.6)

# Настраиваем график
plt.title(f'Корреляция между длинной объявления и конверсией в подачу заявки (r = {correlation:.2f})')
plt.xlabel('Description Length')
plt.ylabel('Apply Conversion')
plt.grid(True)

# Показать график
plt.tight_layout()
plt.show()
```

Полученный график выглядит так



Вывод:

Из графика и низкого коэффициента корреляции видим, что конверсия объявления из просмотра в подачу заявки и длина объявления практически не связаны друг с другом.

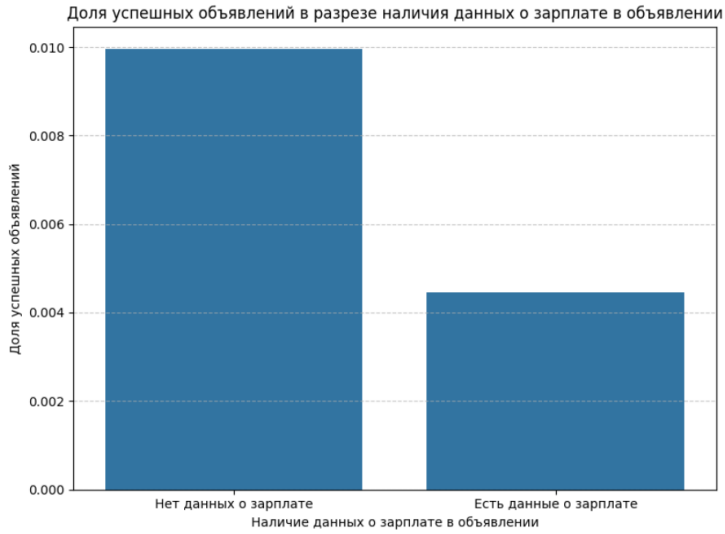
Гипотеза 3 не подтвердилась.

Гипотеза 4 - Объявления с указанной информацией о зарплате чаще бывают успешными

Для проверки этой гипотезы вычислим долю успешных объявлений (успехом считаем закрытое объявление) от всех в разрезе наличия/отсутствия в нем информации о зарплате

```
# Рассчитываем долю строк, где closed_time не null, для каждой категории hourly_rate_available
summary = postings.groupby('hourly_rate_available').apply(
    lambda group: group['closed_time'].notnull().mean()
).reset_index(name='closed_time_ratio')

# Построение столбчатой диаграммы
plt.figure(figsize=(8, 6))
sns.barplot(data=summary, x='hourly_rate_available', y='closed_time_ratio', palette='viridis')

# Настройки графика
plt.title('Доля успешных объявлений в разрезе наличия данных о зарплате в объявлении')
plt.xlabel('Наличие данных о зарплате в объявлении')
plt.ylabel('Доля успешных объявлений')
plt.xticks([0, 1], ['Нет данных о зарплате', 'Есть данные о зарплате'])


plt.grid(axis='y', linestyle='--', alpha=0.7)

# Показать график
plt.tight_layout()
plt.show()
```

Полученный график выглядит так



Вывод:

Видим, что доля успешных объявлений, напротив, выше для объявлений без указания данных о зарплате

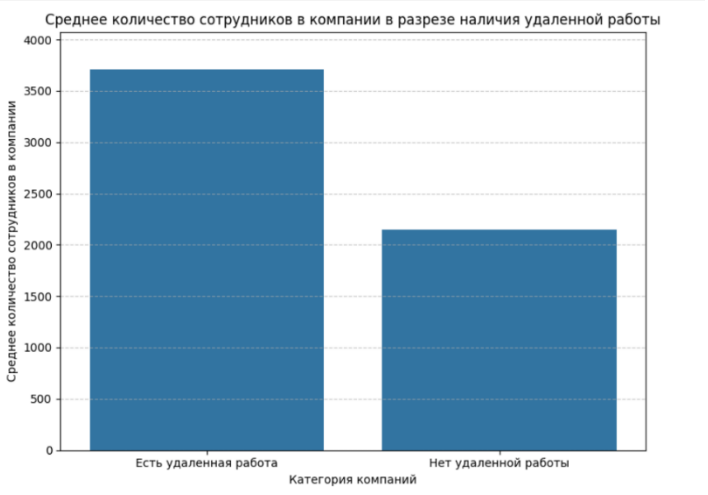
Гипотеза 4 не подтвердилась.

Гипотеза 5 - В среднем, в компаниях, которые разрешают удаленную работу, больше сотрудников, чем в тех, которые не разрешают

Для проверки этой гипотезы посчитаем среднее количество сотрудников в следующем разрезе: компании, выложившие хотя бы одно объявление с доступной удаленной работой, отнесем к разрешающим удаленную работу, остальные – к запрещающим.

```
# Группировка по company_id
grouped = postings.groupby('company_id').agg(
    has_remote_allowed=('remote_allowed', 'any'),  # Проверяем, есть ли хотя бы одно значение True
    employee_count=('employee_count', 'max')  # Максимальное значение employee_count
).reset_index()

# Создаем категорию для компаний: есть ли удаленная работа
grouped['remote_category'] = grouped['has_remote_allowed'].map(
    {True: 'Есть удаленная работа', False: 'Нет удаленной работы'}
)

# Группировка данных для построения графика
plot_data = grouped.groupby('remote_category')['employee_count'].mean().reset_index()

# Построение столбчатой диаграммы
plt.figure(figsize=(8, 6))
sns.barplot(data=plot_data, x='remote_category', y='employee_count')

# Настройки графика
plt.title('Среднее количество сотрудников в компании в разрезе наличия удаленной работы')
plt.xlabel('Категория компаний')
plt.ylabel('Среднее количество сотрудников в компании')
plt.ylim(0, plot_data['employee_count'].max() * 1.1)  # Добавляем небольшой отступ сверху
plt.grid(axis='y', linestyle='--', alpha=0.7)

# Показать график
plt.tight_layout()
plt.show()
```

Полученный график выглядит так



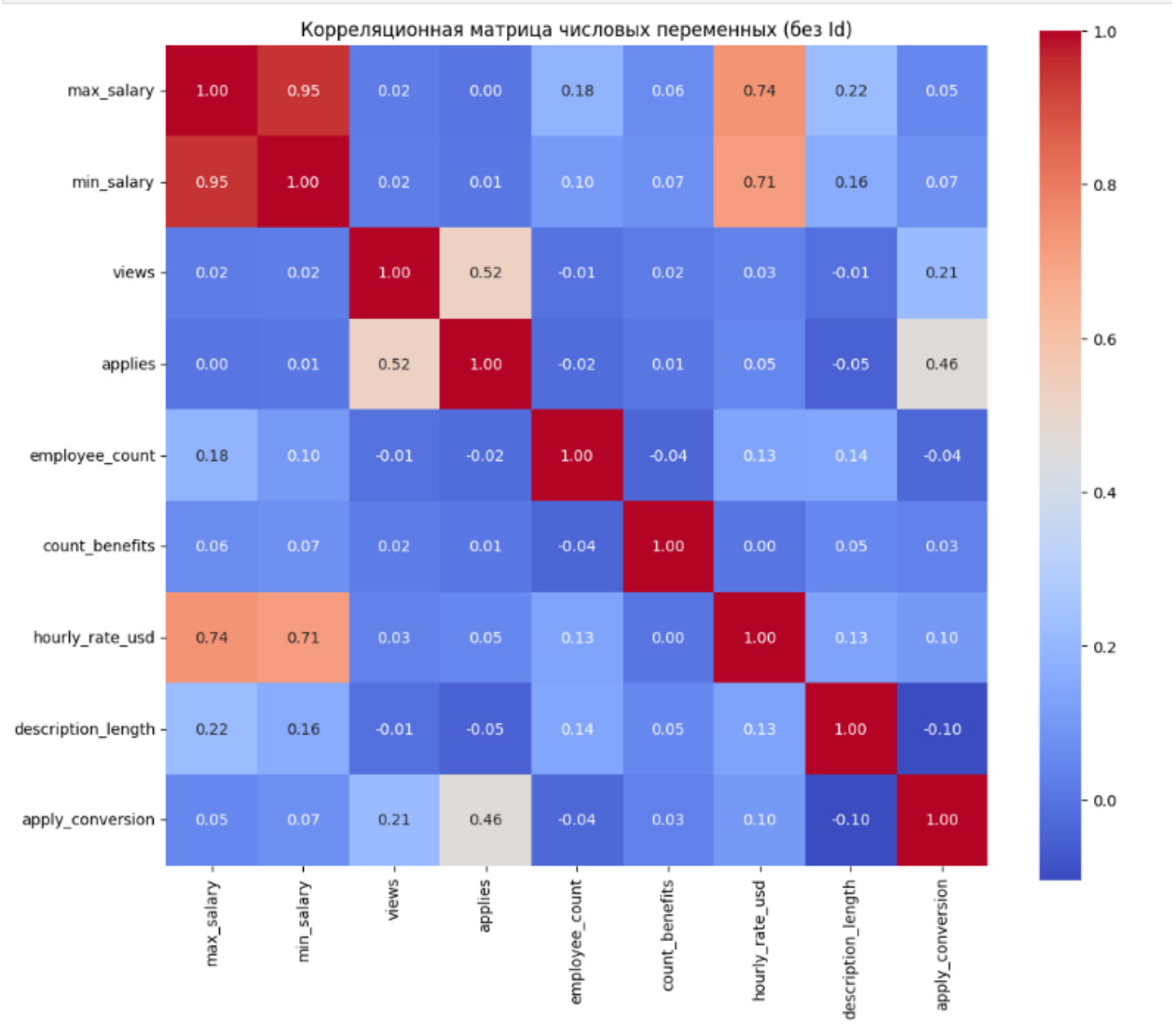
Вывод:

Видим, что в компаниях, предоставляющих возможность работать удаленно, действительно в среднем больше сотрудников.

Гипотеза 5 – подтверждена.

Анализ корреляции

```
# Исключаем столбцы ID (например, job_id и company_id) перед расчетом корреляции
columns_to_exclude = ['job_id', 'company_id']
columns_for_corr = postings.drop(columns=columns_to_exclude, errors='ignore').select_dtypes(include=['float64', 'int64'])

# Расчет корреляционной матрицы
correlation_matrix = columns_for_corr.corr()

# Построение тепловой карты
plt.figure(figsize=(12, 10))
sns.heatmap(correlation_matrix, annot=True, fmt=".2f", cmap='coolwarm', cbar=True, square=True)

# Настройки графика
plt.title('Корреляционная матрица числовых переменных (без Id)')
plt.show()
```



Выводы корреляционного анализа:

1) Значимая корреляция (помимо столбцов, зависящих друг от друга вследствие методики их расчета) есть только между количеством просмотров объявления и количеством поданных заявок

ЗАКЛЮЧЕНИЕ

В ходе научно-исследовательской работы был проведен анализ базы данных «LinkedIn Job Postings (2023 - 2024)», сделаны выводы, которые могут помочь как .соискателям при выборе работы, так и работодателям при размещении объявлений.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1) БУРНАШЕВ Р.А. Методические указания по программному обеспечению «Pandas». - 1 изд. - Казань: КФУ, 2022. - 25 с.

2) Методические указания по программному обеспечению «Seaborn» // [Электронный ресурс]: seaborn.pydata URL: https://seaborn.pydata.org/tutorial.html (дата обращения: 14.09.2024).

3) Методические указания по программному обеспечению «Plotly» // [Электронный ресурс]: habr.com URL: https://habr.com/ru/companies/skillfactory/articles/506974/ (дата обращения: 15.09.2024).

4) «Pandas. Работа с данными» // [Электронный ресурс]: vk.com URL: https://vk.com/wall-159224823_92912 (дата обращения: 15.09.2024).

5) «Python. Визуализация данных: Matplotlib, Seaborn, Mayavi» // [Электронный ресурс]: vk.com URL: https://vk.com/wall-192648009_320 (дата обращения: 15.09.2024).