Университет ИТМО

Практическая работа №2

по дисциплине «Визуализация и моделирование»

Автор: Власов Матвей Иванович
Поток: ВИМ 1.1
Группа: K3240
Факультет: ИКТ
Преподаватель: Чернышева А. В.

Санкт-Петербург, 2021 г.

Датасет

Для дальнейшей работы выбран датасет: "Russian Presidental Elections 2018 Voting Data" (https://www.kaggle.com/valenzione/russian-presidental-elections-2018-voting-data)

Описание датасета

В нашем датасете содержится информация об итогах выборов 2018 года, полученная с официального сайта ЦИК РФ.

Названия большинства стобцов исходного датасета представим в графе "Описание а сами названия сократим для удобства в дальнейшем:

Столбец	Описание	Тип	Шкала
PS_ID	Идентификатор избирательного участка	INT	Относительная
REGION	Название региона	STRING	Номинальная
SUBREGION	Название округа	STRING	Номинальная
N_ALL	Число избирателей, включенных в список избирателей	INT	Относительная
N_GIVEN	Число избирательных бюллетеней, полученных участковой избирательной комиссией	INT	Относительная
N_EARLY	Число избирательных бюллетеней, выданных избирателям, проголосовавшим досрочно	INT	Относительная
N_IN	Число избирательных бюллетеней, выданных в помещении для голосования в день голосования	INT	Относительная
N_OUT	Число избирательных бюллетеней, выданных вне помещения для голосования в день голосования	INT	Относительная
N_LEFT	Число погашенных избирательных бюллетеней	INT	Относительная
N_PORTABLE	Число избирательных бюллетеней в переносных ящиках для голосования	INT	Относительная
N_STATIC	Число бюллетеней в стационарных ящиках для голосования	INT	Относительная
N_INVALID	Число недействительных избирательных бюллетеней	INT	Относительная
N_VALID	Число действительных избирательных бюллетеней	INT	Относительная
N_LOST	Число утраченных избирательных бюллетеней	INT	Относительная
N_UNUSED	Число избирательных бюллетеней, не учтенных при получении	INT	Относительная
BABURIN	Бабурин Сергей Николаевич	INT	Относительная
GRUDININ	Грудинин Павел Николаевич	INT	Относительная
ZHIRINOVSKY	Жириновский Владимир Вольфович	INT	Относительная
PUTIN	Путин Владимир Владимирович	INT	Относительная
SOBCHAK	Собчак Ксения Анатольевна	INT	Относительная
SURAYKIN	Сурайкин Максим Александрович	INT	Относительная
TITOV	Титов Борис Юрьевич	INT	Относительная
YAVLINSKY	Явлинский Григорий Алексеевич	INT	Относительная

Задачи, решаемые при помощи датасета

1. Визуализация результатов выборов.
2. Анализ данных на предмет возможных фальсификаций.
3. Выявление особенностей голосования в различных регионах.

Работа с датасетом

Обратите внимание, что часть кода и таблиц отображается неполностью.
Для просмотра недостающей информации откройте файл с исходным кодом.

Отметим, что данные нашего датасета немного не совпадают с данными на сайте ЦИК. Расхождения замечены в нескольких регионах. Как мы это учтём: 1. При анализе выборов по регионам заменим часть датасета на данные с сайта ЦИК там, где замечены несовпадения 2. При дальнейшем анализе данных регионов по участкам будем иметь в виду, что наша информация немного неполная/неактуальная

In [1]:
```
import matplotlib.pyplot as plt
import numpy as np
import pandas as pd
```

In [2]:
```
df = pd.read_csv('voting_data.csv')
df
```

Out[2]:

	PS_ID	REGION	SUBREGION	N_ALL	N_GIVEN	N_EARLY	N_IN	N_OUT	N_LEI
0	8140	98 Город Байконур (Республика Казахстан)	98 Город Байконур (Республика Казахстан)	2132	2000	0	1447	11	5
1	8141	98 Город Байконур (Республика Казахстан)	98 Город Байконур (Республика Казахстан)	2207	2000	0	1470	14	5
2	8142	98 Город Байконур (Республика Казахстан)	98 Город Байконур (Республика Казахстан)	2249	2000	0	1490	7	5
3	8143	98 Город Байконур (Республика Казахстан)	98 Город Байконур (Республика Казахстан)	1769	1500	0	1065	48	3
4	8144	98 Город Байконур (Республика Казахстан)	98 Город Байконур (Республика Казахстан)	1880	1500	0	1171	13	3
...	...	...	...	...	...	...	...	...	
97689	1310	Ямало-Ненецкий автономный округ	13 Ямальская	2892	2900	1000	1891	0	
97690	1311	Ямало-Ненецкий автономный округ	13 Ямальская	2867	2900	921	1941	0	
97691	1312	Ямало-Ненецкий автономный округ	13 Ямальская	2895	2900	927	1964	0	
97692	1313	Ямало-Ненецкий автономный округ	13 Ямальская	3397	3450	1103	2281	0	
97693	1314	Ямало-Ненецкий автономный округ	13 Ямальская	2666	3000	1022	1632	0	3

97694 rows × 23 columns

Добавим в таблицу информацию об общем количестве голосов на участке

In [3]:
```
voted_list = []

for _, row in df.iterrows():
    voted_list.append(row['N_EARLY'] + row['N_IN'] + row['N_OUT']) # or 

df['N_VOTED'] = voted_list
df
```

Out[3]:

	PS_ID	REGION	SUBREGION	N_ALL	N_GIVEN	N_EARLY	N_IN	N_OUT	N_LE
0	8140	98 Город Байконур (Республика Казахстан)	98 Город Байконур (Республика Казахстан)	2132	2000	0	1447	11	5
1	8141	98 Город Байконур (Республика Казахстан)	98 Город Байконур (Республика Казахстан)	2207	2000	0	1470	14	5
2	8142	98 Город Байконур (Республика Казахстан)	98 Город Байконур (Республика Казахстан)	2249	2000	0	1490	7	5
3	8143	98 Город Байконур (Республика Казахстан)	98 Город Байконур (Республика Казахстан)	1769	1500	0	1065	48	3
4	8144	98 Город Байконур (Республика Казахстан)	98 Город Байконур (Республика Казахстан)	1880	1500	0	1171	13	3
...	...	...	...	...	...	...	...	...	
97689	1310	Ямало-Ненецкий автономный округ	13 Ямальская	2892	2900	1000	1891	0	
97690	1311	Ямало-Ненецкий автономный округ	13 Ямальская	2867	2900	921	1941	0	
97691	1312	Ямало-Ненецкий автономный округ	13 Ямальская	2895	2900	927	1964	0	
97692	1313	Ямало-Ненецкий автономный округ	13 Ямальская	3397	3450	1103	2281	0	
97693	1314	Ямало-Ненецкий автономный округ	13 Ямальская	2666	3000	1022	1632	0	3

97694 rows × 24 columns

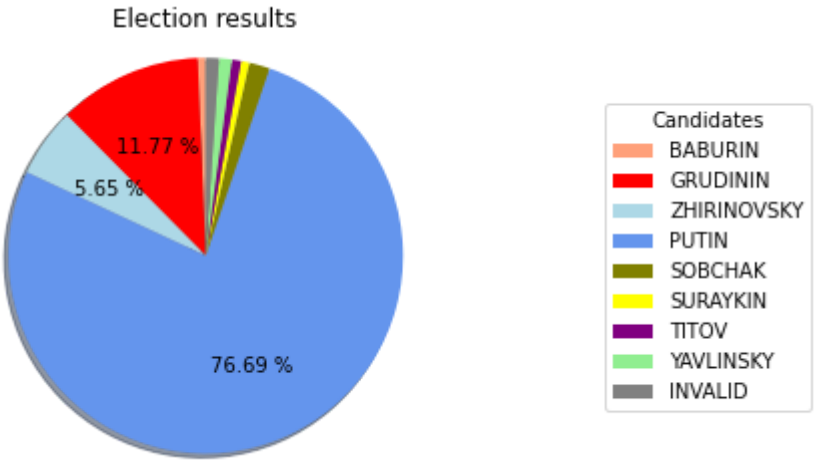
Объявим некоторые константы, которые пригодятся при анализе данных

In [4]:
```
start = list(df.columns).index('BABURIN')
end = list(df.columns).index('YAVLINSKY') + 1
CANDS = [i for i in df.columns[start:end]]
```

```
COLORS = ['lightsalmon', 'red', 'lightblue', 'CornflowerBlue', 'olive',
DF_LIST = [i for i in range(df.shape[0])]
INCOMPLETE_REGIONS = ['Ханты-Мансийский автономный округ - Югра', 'Респу
                      'Чувашская Республика - Чувашия', 'Брянская область', 'Во
                      'Калининградская область', 'Кемеровская область', 'Магада
                      'Нижегородская область', 'город Москва', 'Территория за п
```

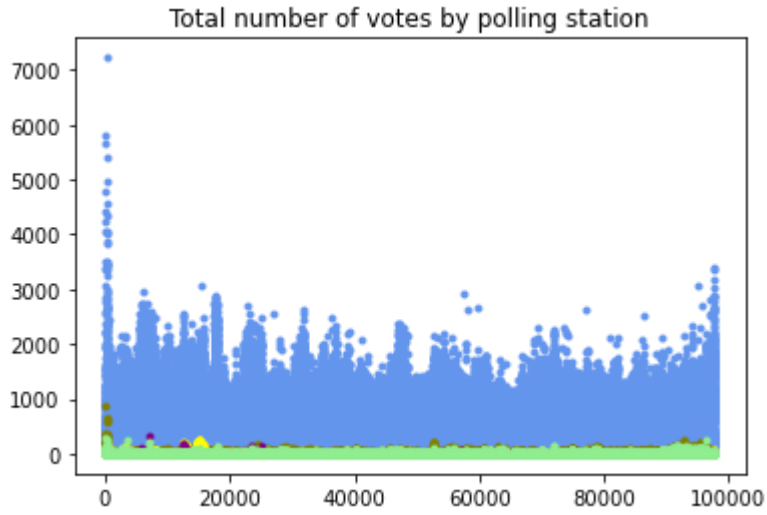
Представим результаты выборов на круговой диаграмме

In [5]:
```
labels = CANDS.copy() + ['INVALID']
start = list(df.columns).index('BABURIN')
end = list(df.columns).index('YAVLINSKY') + 1
sizes = [df[i].sum() for i in df.columns[start:end]]
sizes.append(df['N_INVALID'].sum() + df['N_LOST'].sum())

_, ax = plt.subplots()
ax.pie(sizes, labels=labels,
        autopct=(lambda x: f'{x:.2f} %' if x > 5.0 else ''),
        shadow=True, startangle=90, labeldistance=None,
        colors=COLORS + ['grey'])
ax.axis('equal')
ax.legend(labels, title="Candidates", loc="center", bbox_to_anchor=(1, 0

plt.title('Election results')
plt.show()
```



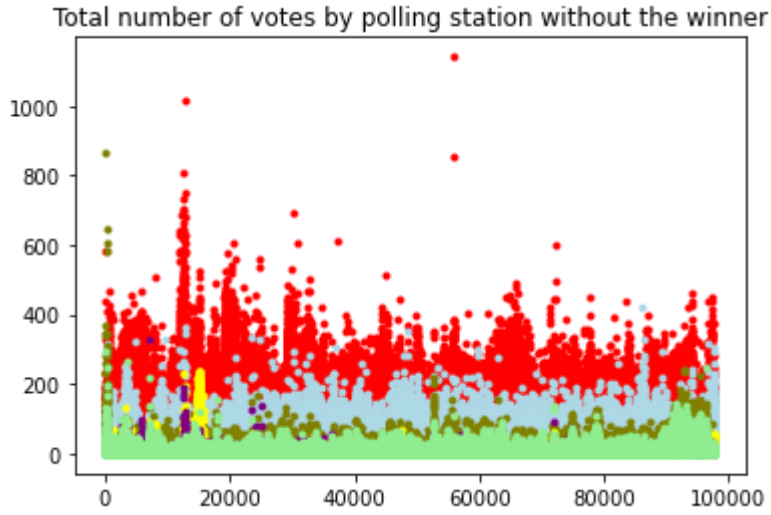
На диаграмме мы видим распределение голосов на выборах. Отметим, что показанные на диаграмме проценты соответствуют официальным данным (наши недостающие данных составляют слишком малую часть, поэтому на общие итоги не влияют)

In [6]:
```
for k in range(len(CANDS)):
    plt.plot(DF_LIST, [df[CANDS[k]][i] for i in range(df.shape[0])], '.'

plt.title('Total number of votes by polling station')
plt.show()
```



На диаграмме выше представлено количество голосов за каждого кандидата по УИК. Как мы видим, за Путина в среднем голосовало по несколько тысяч человек на участке, в то время как за других кандидатов - явно меньше тысячи. Чтобы подробнее увидеть информацию о других кандидатах, уберём голоса за победителя

In [7]:
```
for k in range(len(CANDS)):
    if k == CANDS.index('PUTIN'): continue
    plt.plot(DF_LIST, [df[CANDS[k]][i] for i in range(df.shape[0])], '.'

plt.title('Total number of votes by polling station without the winner')
plt.show()
```



По данной диаграмме видно, что на втором месте с достаточным отрывов находится Грудинин. В районе 17000-х участков можно заметить резкий рост голосов за Сурайкина. Возможно, в дальнейшем остановимся на этом подробнее Заметим, что по анализу отдельных УИК тяжело сделать какие-либо выводы, кроме самых очевидных. Будем проводить анализ по регионамФункция удаления цифр нужна для двух регионов, названия которых в датасете начинаются с цифры. Для удобства эти цифры удалим

In [8]:
```
def remove_digits(text):
    if text[0].isdigit():
        temp = list(map(lambda x: '' if x.isdigit() else x, text))
        text = ''.join(temp).strip()
    return text
```

Создадим таблицу с данным по регионам. Для этого просуммируем данные из всех участков данного региона и запишем их в одну строку. Для регионов, где в первой таблице содержатся неполные данные, возьмём данные с сайта ЦИК

In [9]:
```
df_regions = pd.DataFrame(columns=df.columns)
df_regions = df_regions.drop(columns=['PS_ID', 'SUBREGION', 'N_VOTED'])

for region in df['REGION'].unique():
    info = {}
    new_region = remove_digits(region)

    info['REGION'] = new_region
    for col in df.columns[3:-1]:
        info[col] = df[df['REGION'] == region][col].sum()
    df_regions = df_regions.append(info, ignore_index=True)

for region in INCOMPLETE_REGIONS:
    new_info = []
    data = pd.read_csv(region + '.csv')
    for row in data['Unnamed: 2']:
        try:
            new_info.append(int(row))
        except ValueError:
            continue

    index = int(df_regions[df_regions['REGION'] == region].index.values)
    for i, col in enumerate(df_regions.columns[1:]):
        df_regions.at[index, col] = new_info[i+2]
```

In [10]:
```
df_regions.to_csv('regions_data.csv')
DF_REGIONS_LIST = [i for i in range(df_regions.shape[0])]
```

Добавим в таблицу данные об общем количестве проголосовавших и явке

In [11]:
```
turnout_list = []
voted_list = []

for _, row in df_regions.iterrows():
    voted = row['N_EARLY'] + row['N_IN'] + row['N_OUT']
    voted_list.append(voted)
    turnout_list.append(voted/row['N_ALL'])

df_regions['N_VOTED'] = voted_list
df_regions['TURNOUT'] = turnout_list
df_regions.to_csv('regions_data.csv')
df_regions
```

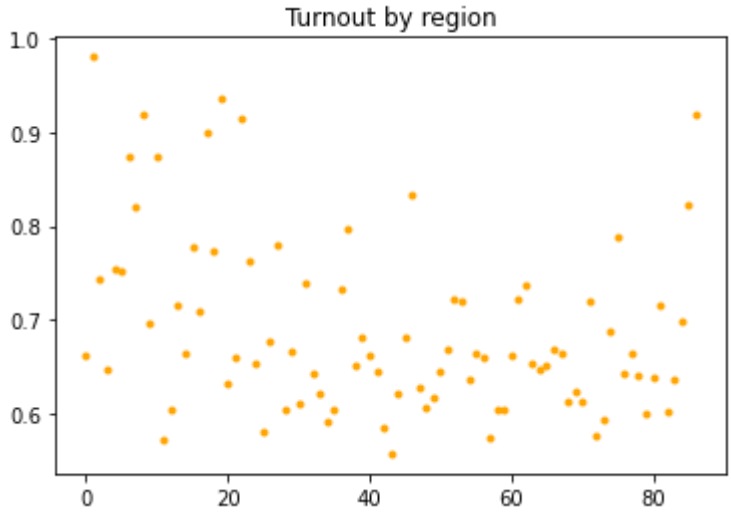
Out[11]:

	REGION	N_ALL	N_GIVEN	N_EARLY	N_IN	N_OUT	N_LEFT	N_PORTABLE
0	Город Байконур (Республика Казахстан)	14575	12800	0	9491	168	3141	168
1	Территория за пределами РФ	483957	1408383	53482	403108	18026	933738	71484
2	Республика Адыгея (Адыгея)	336720	328666	0	235040	15183	78443	15182
3	Республика Алтай	158891	156580	1186	97103	4630	53661	5816
4	Республика Башкортостан	3045698	2866878	0	2186576	111258	569036	111248
...	...	...	...	...	...	...	...	...
82	Еврейская автономная	128844	120568	106	73398	4123	42941	4229

	область							
83	Ненецкий автономный округ	39470	33688	6116	18414	579	8579	6695
84	Ханты-Мансийский автономный округ - Югра	1130343	1095884	26633	745638	15804	307809	42426
85	Чукотский автономный округ	33541	32298	2531	24193	874	4700	3405
86	Ямало-Ненецкий автономный округ	370823	372171	41135	295648	4008	31380	45143

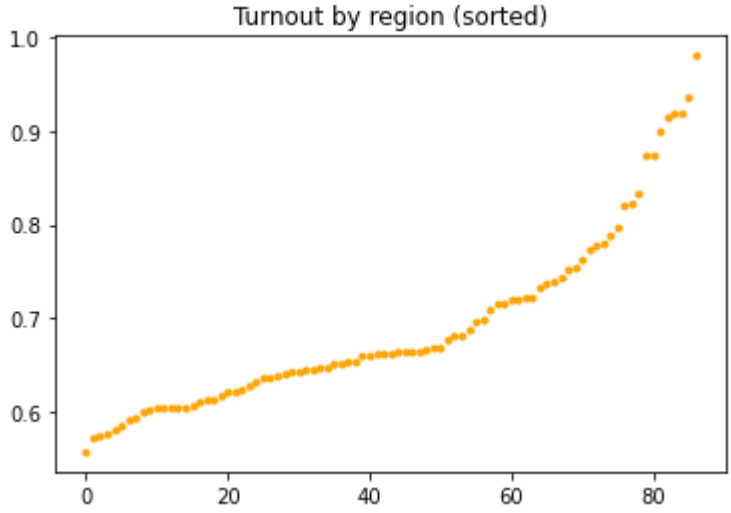
87 rows × 23 columns

In [12]:
```
plt.plot(DF_REGIONS_LIST, turnout_list, '.', color='orange')
plt.title('Turnout by region')
plt.show()
```



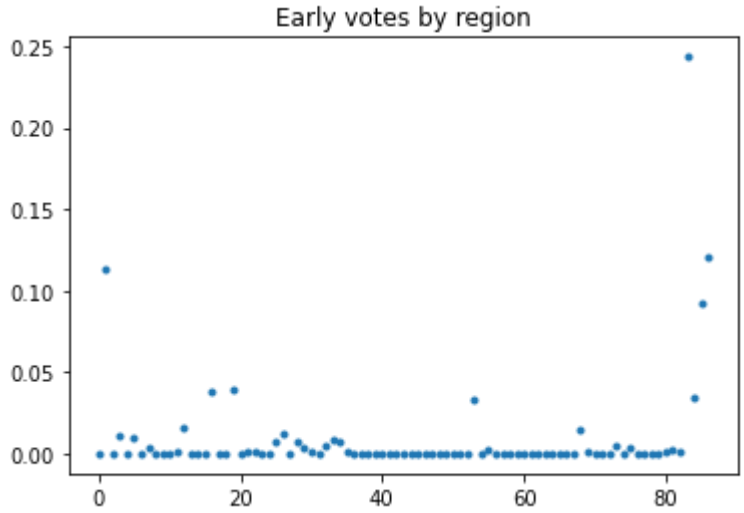
На графике показано распределение явки по регионам. Видно, что во всех регионах явка больше половину, в некоторых - почти 100 % (к этим регионам ещё вернёмся)

In [13]:
```
turnout_list = [df_regions['N_VOTED'][i]/df_regions['N_ALL'][i] for i in
plt.plot(DF_REGIONS_LIST, sorted(turnout_list), '.', color='orange')
plt.title('Turnout by region (sorted)')
plt.show()
```



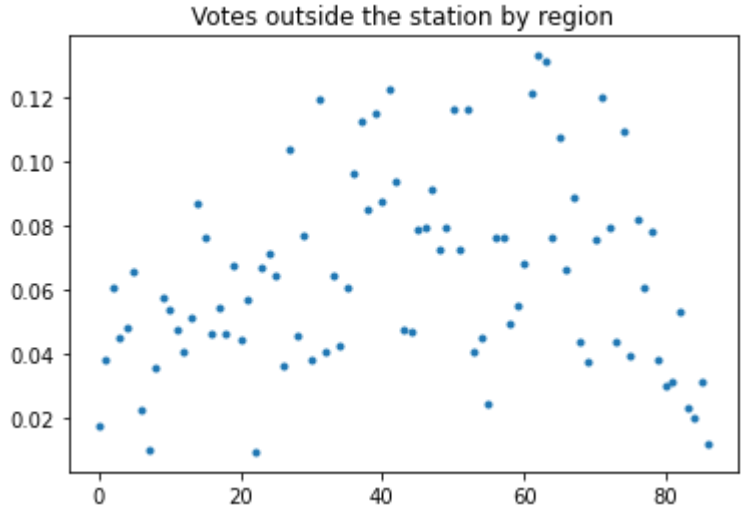
Для удобства отсортировали график явки по регионам. Видим, что в большинстве регионов явка составила около 65 %

In [14]:
```
plt.plot(DF_REGIONS_LIST, [df_regions['N_EARLY'][i]/df_regions['N_VOTED'
plt.title('Early votes by region')
plt.show()
```



На графике выше - процент голосов до дня голосования. В среднем он почти нулевой, однако есть регионы, где он составляет 5, 10 и даже 25 процентов

In [15]:
```
plt.plot(DF_REGIONS_LIST, [df_regions['N_OUT'][i]/df_regions['N_VOTED'][
plt.title('Votes outside the station by region')
plt.show()
```

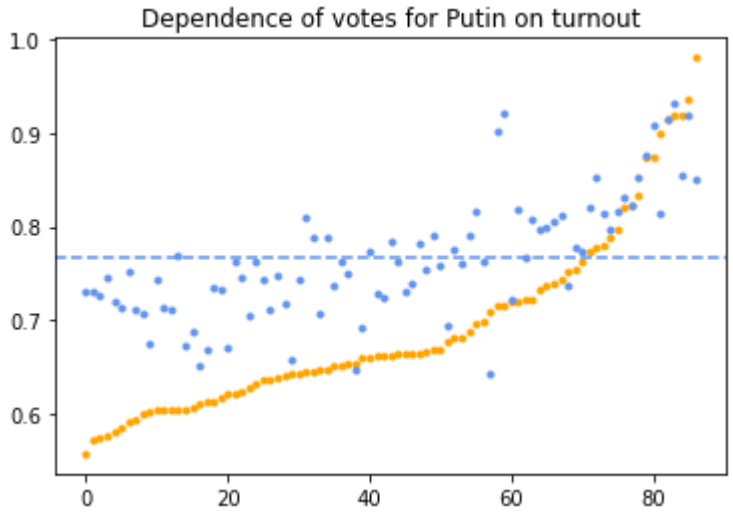


На данном графике - процент голосов за пределами УИК. Видим, что точки на графике распредены почти равномерно

Добавим функцию для изображения линии, равной среднему проценту победителя в выборах. Будем её использовать во многих графиках

In [16]:
```
def add_winner_average_line(obj):
    putin_list = [df_regions['PUTIN'][i]/df_regions['N_VOTED'][i] for i
    obj.axhline(y=sum(putin_list)/len(putin_list), ls='--', color=COLORS
```

In [17]:
```
putin_list = [df_regions['PUTIN'][i]/df_regions['N_VOTED'][i] for i in r

turnout_sorted, putin_sorted = zip(*sorted(zip(turnout_list, putin_list)
plt.plot(DF_REGIONS_LIST, turnout_sorted, '.', color='orange')
plt.plot(DF_REGIONS_LIST, putin_sorted, '.', color=COLORS[CANDS.index('P

add_winner_average_line(plt)
plt.title('Dependence of votes for Putin on turnout')
plt.show()
```



На графике представлена зависимость голосов за Путина от явки. Как мы видим, наибольший процент голосов за Путина - в регионах с максимальной явкой. Более того, среди регионов с явкой меньше 75 % нет почти ни одного, где процент голосов за Путина превышал бы средний. Подозрительно. Далее изучим подробнее регионы с самой большой явкой (и низкой тоже). Для этого отсортируем нашу таблицу по явке

In [18]:
```
df_regions_sorted = df_regions.sort_values(by='TURNOUT', ascending=False
df_regions_sorted
```

Out[18]:

	REGION	N_ALL	N_GIVEN	N_EARLY	N_IN	N_OUT	N_LEFT	N_PORTABLE
0	Территория за пределами РФ	483957	1408383	53482	403108	18026	933738	71484
1	Республика Тыва	175102	185961	6476	146485	11046	21951	17370
2	Ямало-Ненецкий автономный округ	370823	372171	41135	295648	4008	31380	45143
3	Кабардино-Балкарская Республика	528431	534343	0	467655	17453	49232	17453
4	Чеченская Республика	709635	695880	500	642850	6250	46222	6750
...	...	...	...	...	...	...	...	...
82	Забайкальский край	790054	755211	3397	425198	29543	297073	32939
83	Тверская область	1070221	990731	0	567381	48898	374450	48895
84	Новгородская область	502905	479754	0	266259	21925	191569	21925
85	Республика Карелия	519667	501054	257	282899	14071	203821	14327
86	Иркутская область	1877547	1763939	178	995467	50075	718218	50251

87 rows × 23 columns

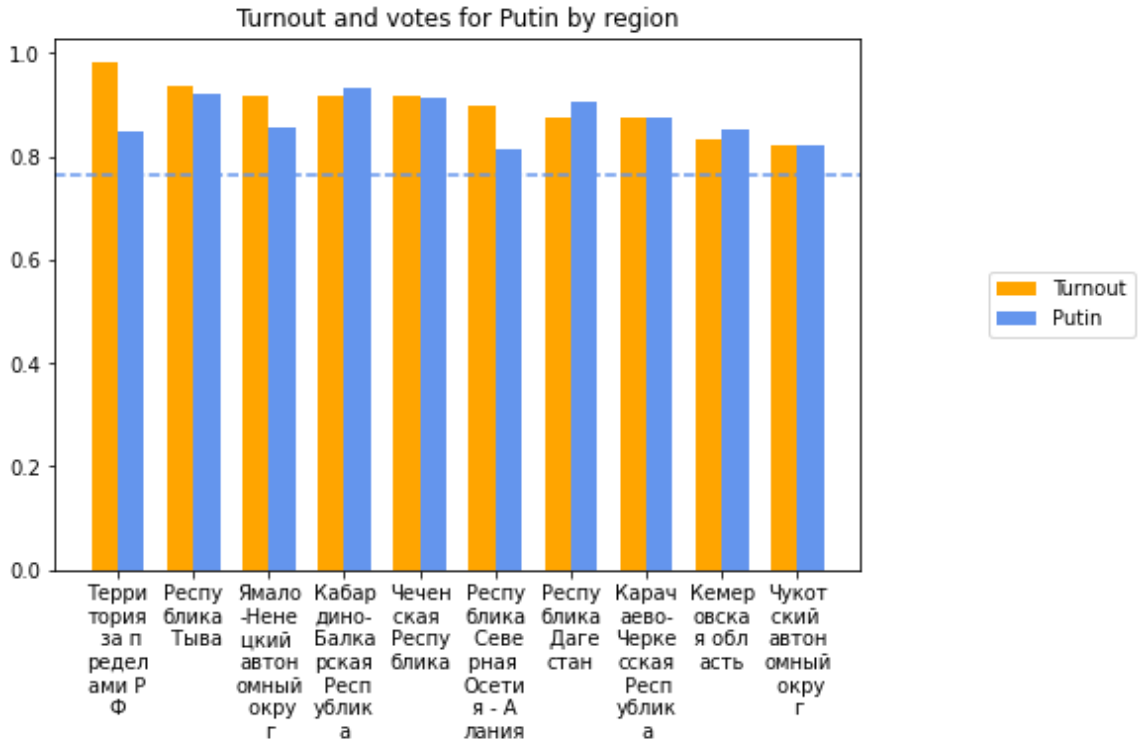
Создадим функцию отображения явки и голосов за Путина в виде столбчатой диаграммы, чтобы подробнее проанализировать интересующие нас регионы

In [19]:

```
def draw_turnout_bar(num, reverse=False):
    if num > 10:
        print('The maxmium number of regions is 10')
        return

    start = 0 if not reverse else df_regions_sorted.shape[0] - 1
    stop = num if not reverse else start - num
    step = 1 if not reverse else -1
    turnout_num_list = [df_regions_sorted['TURNOUT'][i] for i in range(s
    putin_num_list = [df_regions_sorted['PUTIN'][i]/df_regions_sorted['N_
    region_num_list = []
    for i in range(start, stop, step):
        new_region = ''
        region = df_regions_sorted['REGION'][i]
        for j in range(0, len(region), 15 - num):
            new_region += region[j:j + 15 - num] + '\n'
        region_num_list.append(new_region)

    ind = np.arange(len(turnout_num_list))
    width = 0.35

    fig, ax = plt.subplots()
    ax.bar(ind - width/2, turnout_num_list, width,
                    label='Turnout', color='orange')
    ax.bar(ind + width/2, putin_num_list, width,
                    label='Putin', color=COLORS[CANDS.index('PUTIN')])
```

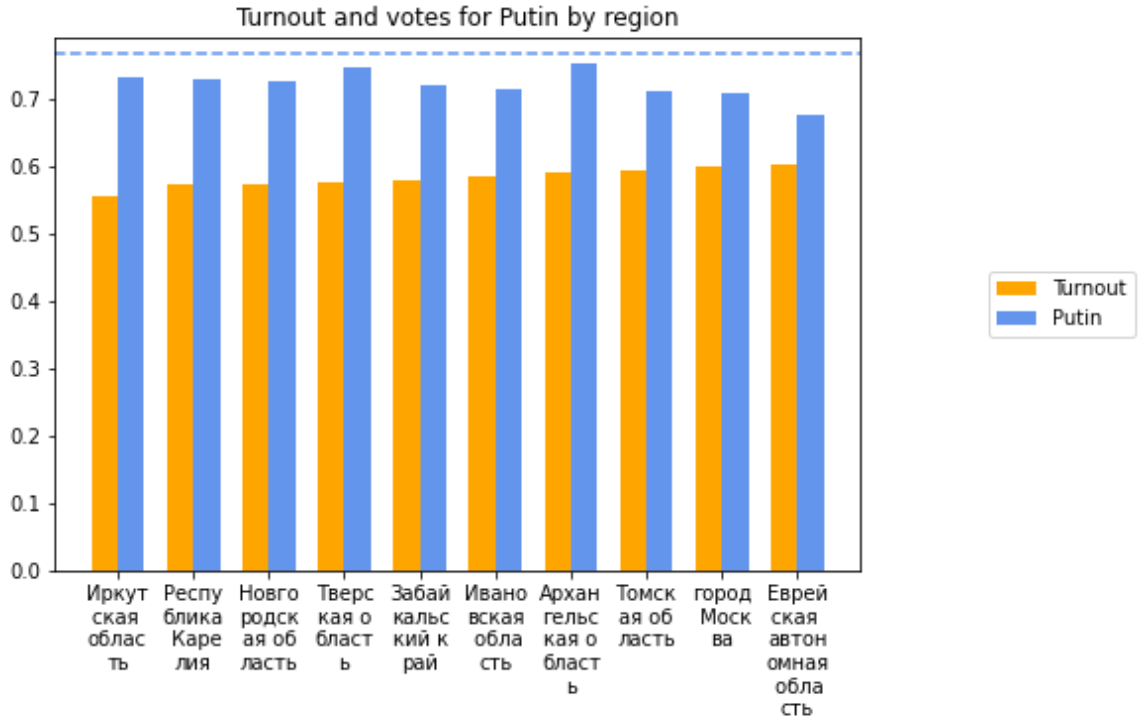
```
    fig.tight_layout(pad=0.1)
    ax.set_title('Turnout and votes for Putin by region')
    ax.set_xticks(ind)
    ax.set_xticklabels(region_num_list, fontdict={'fontsize': 10})
    ax.legend(['Turnout', 'Putin'], loc="center", bbox_to_anchor=(1, 0,
    add_winner_average_line(ax)
```

In [20]:
```
draw_turnout_bar(10)
```



На диаграмме видно, что среди первых десяти регионах по явке нет ни одного, где процент за Путина был бы ниже среднего

In [21]:
```
draw_turnout_bar(10, reverse=True)
```



Здесь же, наоборот, нет ни одного региона, где количество голосов за Путина превышало бы среднее. Что ж, рассмотрим данные регионы ещё подробнееСоздадим функцию для визуализации по региону, которая вызывает ещё три - каждая рисует один график

In [22]:
```
def visualize_region_results(region):
    check_region = region
    if 'Территория' in region:
        check_region = '99 ' + region
    elif 'Казахстан' in region:
        check_region = '98 ' + region
    df_region = df.loc[df['REGION'] == check_region]
    if not len(df_region):
        print('You entered incorrect region')
        return
    if region in INCOMPLETE_REGIONS:
        print('Note that the data about this region is incomplete')
    df_region = df_region.reset_index()
    visualize_turnout(df_region)
    visualize_winner(df_region)
    visualize_dependence(df_region)
```

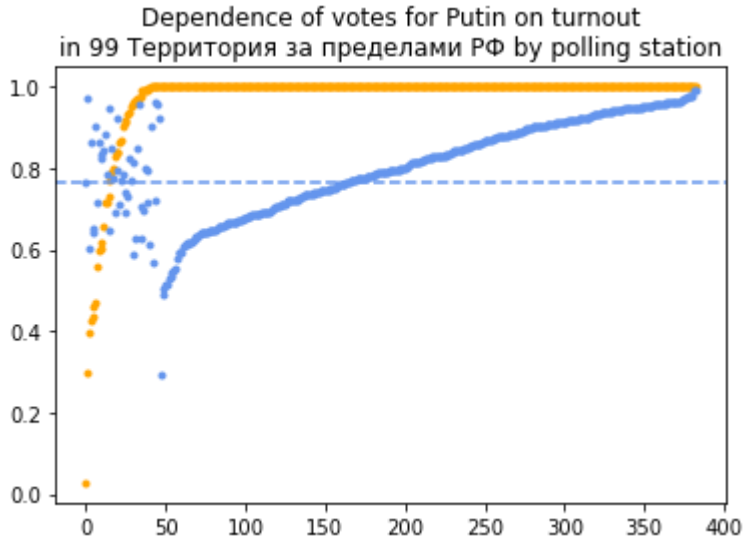
In [23]:
```
def visualize_turnout(df_region):
    plt.plot([i for i in range(df_region.shape[0])], sorted([df_region['
    plt.title(f"Turnout in {df_region['REGION'][0]} by polling station")
    plt.show()
```

In [24]:
```
def visualize_winner(df_region):
    plt.plot([i for i in range(df_region.shape[0])], sorted([df_region['
    plt.title(f"Votes for Putin in {df_region['REGION'][0]} by polling s
    plt.show()
```

In [25]:
```
def visualize_dependence(df_region):
    turnout_list = [df_region['N_VOTED'][i]/df_region['N_ALL'][i] for i 
    putin_list = [df_region['PUTIN'][i]/df_region['N_VOTED'][i] for i in
    turnout_sorted, putin_sorted = zip(*sorted(zip(turnout_list, putin_l
    plt.plot([i for i in range(df_region.shape[0])], turnout_sorted, '.'
    plt.plot([i for i in range(df_region.shape[0])], putin_sorted, '.', 
    add_winner_average_line(plt)

    plt.title(f"Dependence of votes for Putin on turnout\nin {df_region[
    plt.show()
```

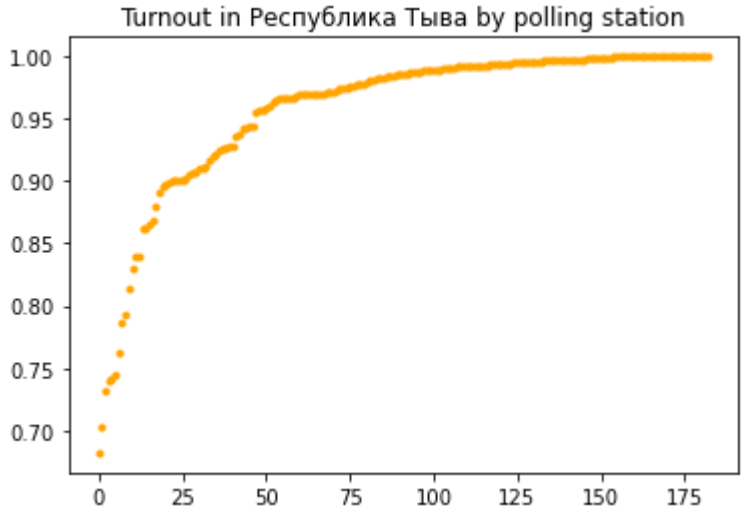
In [26]:
```
visualize_region_results(df_regions_sorted['REGION'][0])
```

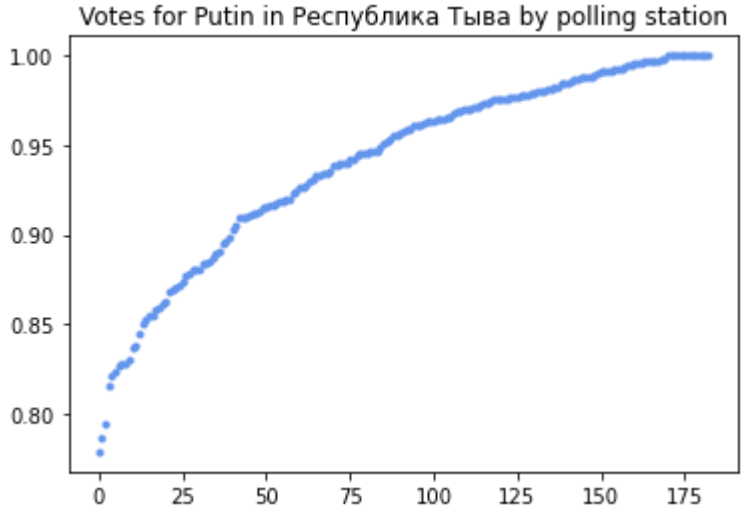
```
Note that the data about this region is incomplete
```

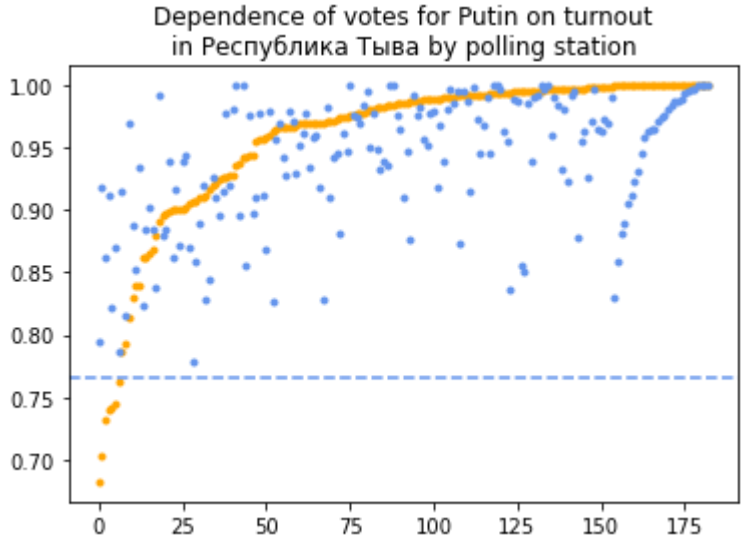


Как мы видим, при голосовании за пределами РФ явка составляет почти 100 % на большинстве участков. Это, скорее всего, связано с тем, что на участках за пределами РФ регистрируются заранее. Есть человек подал заявление на включение в список избирателей, он наверняка сходит и на выборы

In [27]:
```
visualize_region_results(df_regions_sorted['REGION'][1])
```

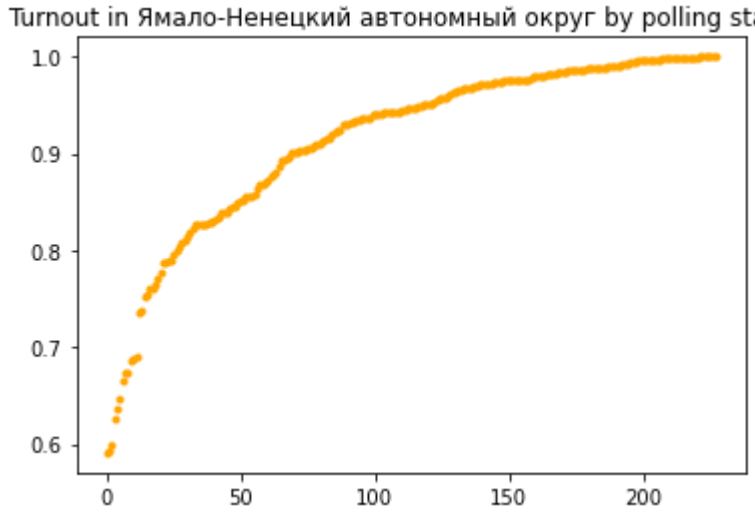


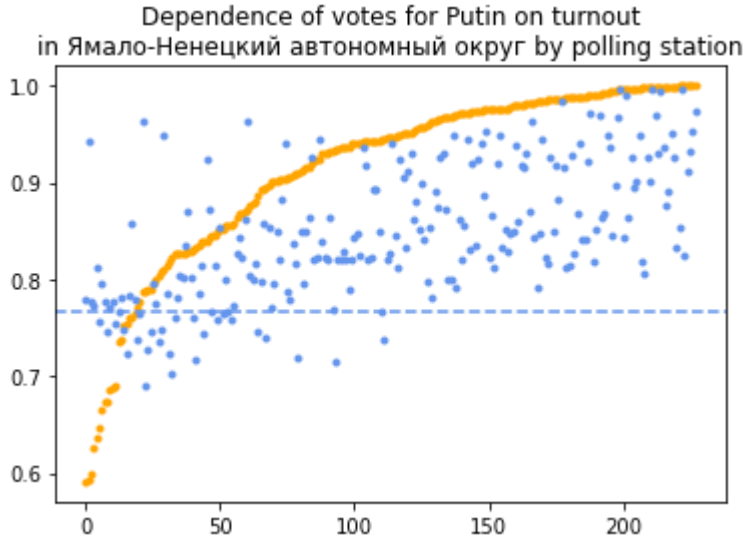




Видим, что даже в пределах региона нет ни одного участка, где процент у Путина был меньше среднего (как, например, на территории за пределами РФ)

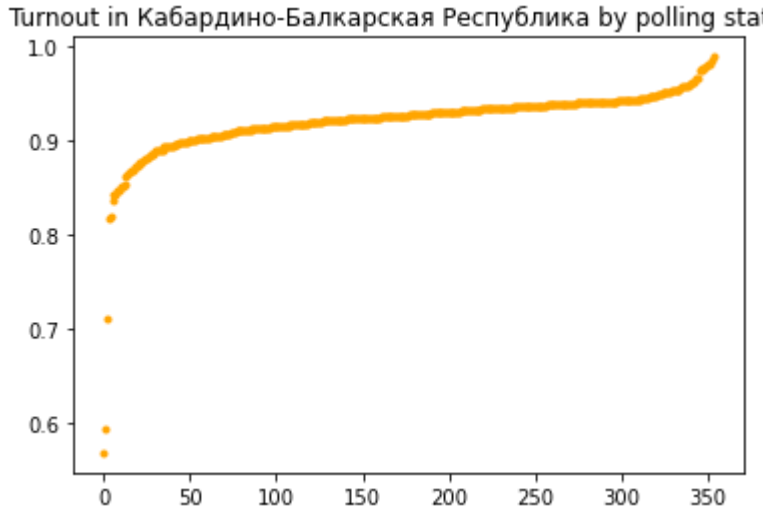
In [28]:
```
visualize_region_results(df_regions_sorted['REGION'][2])
```

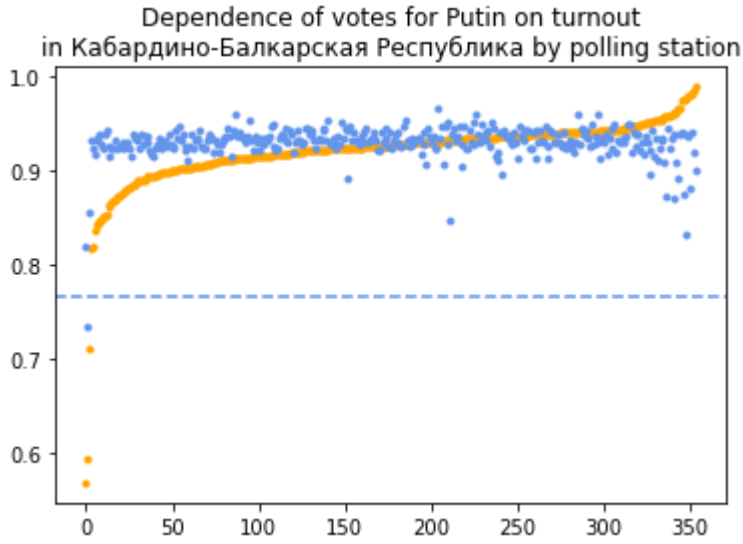




Заметим, что скопление точек у линии среднего значения именно там, где низкая явка

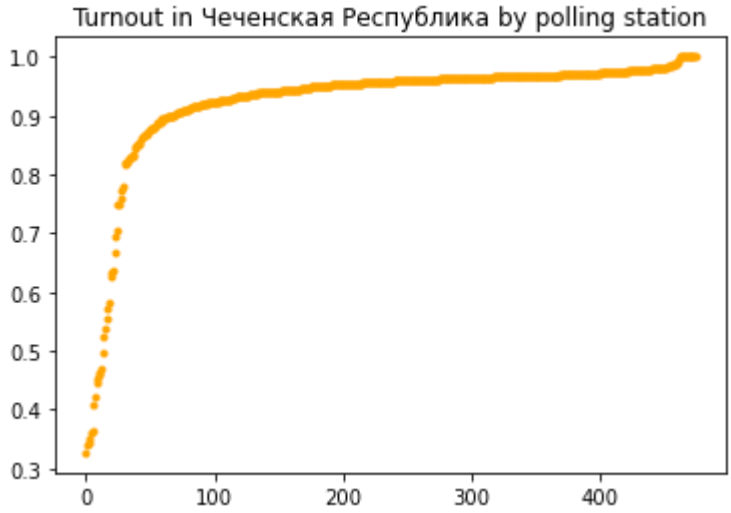
```
In [29]: visualize_region_results(df_regions_sorted['REGION'][3])
```

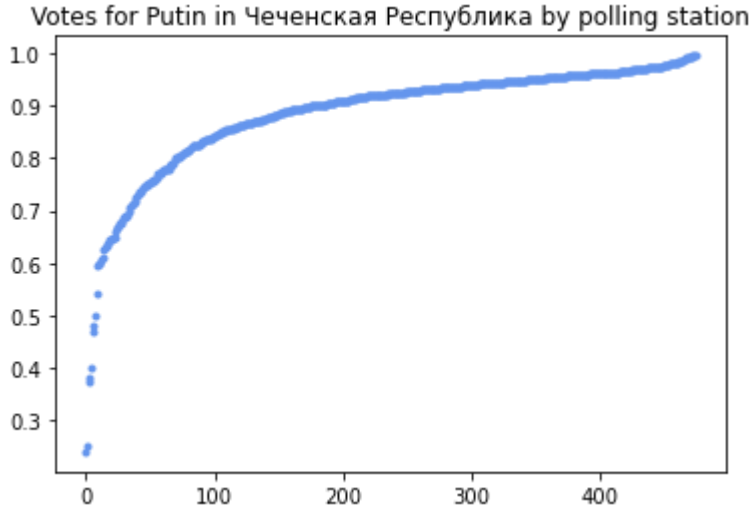


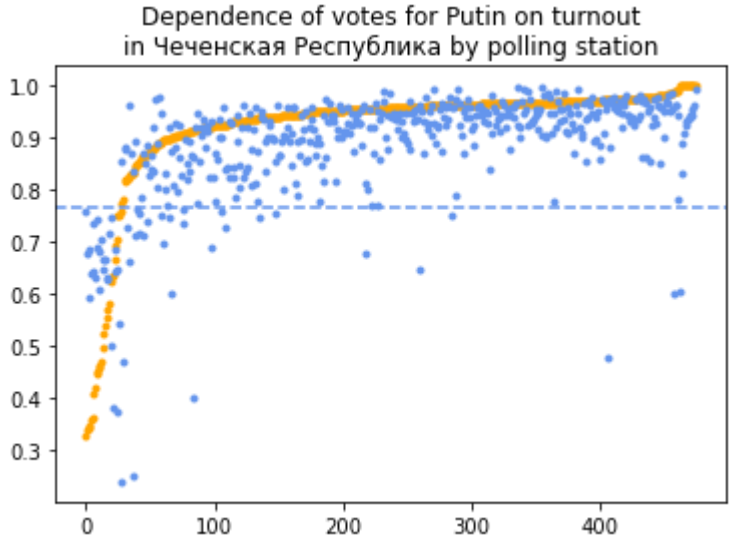


Невероятно высокий процент голосов за Путина вне зависимости от явки (которая, к слову, тоже сосредоточена в районе 90 %)

In [30]:
```
visualize_region_results(df_regions_sorted['REGION'][4])
```

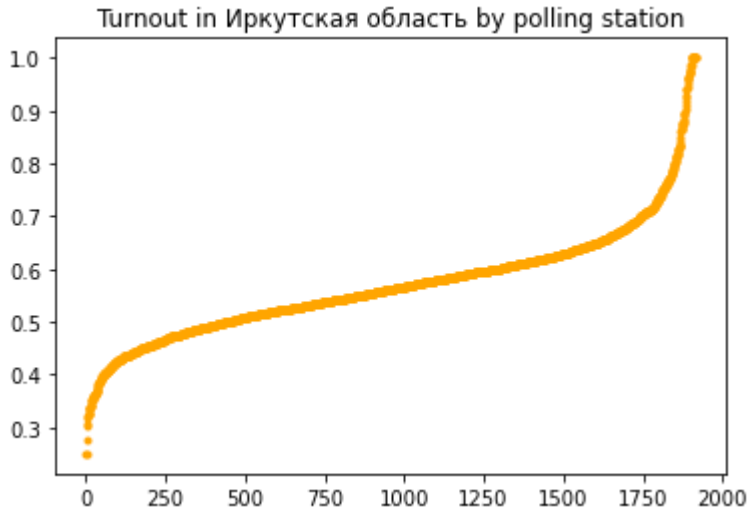


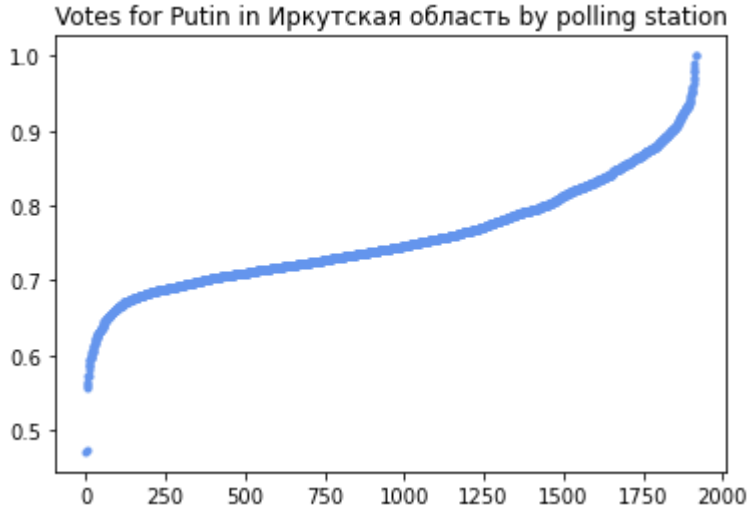


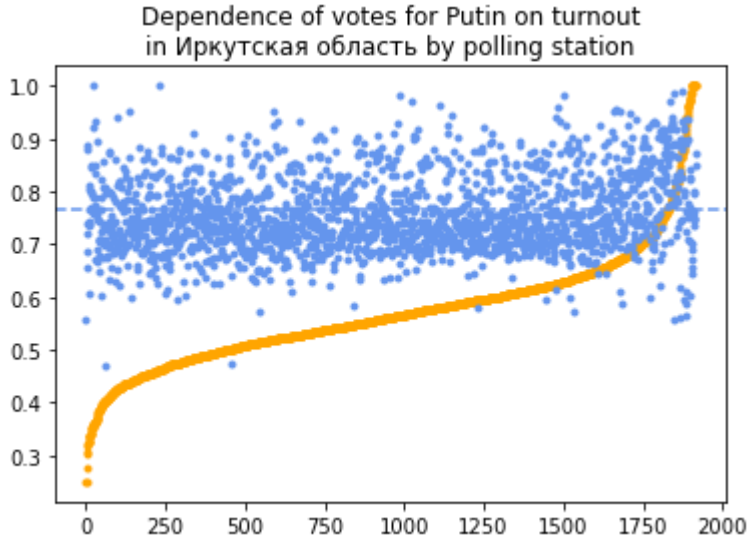


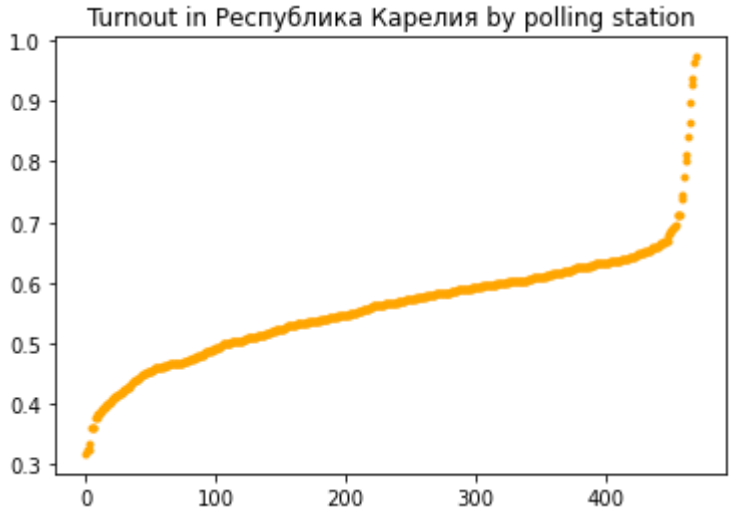
Знакомая ситуация: низкий процент у Путина там же, где низкая явка. И это при том, что на прошлых выборах в данном регионе Путин получил почти 100 % при почти стопроцентной явке. Кажется: в этот раз на нескольких участках были наблюдатели (и испортили всю картину!)Теперь посмотрим на регионы с низкой явкой

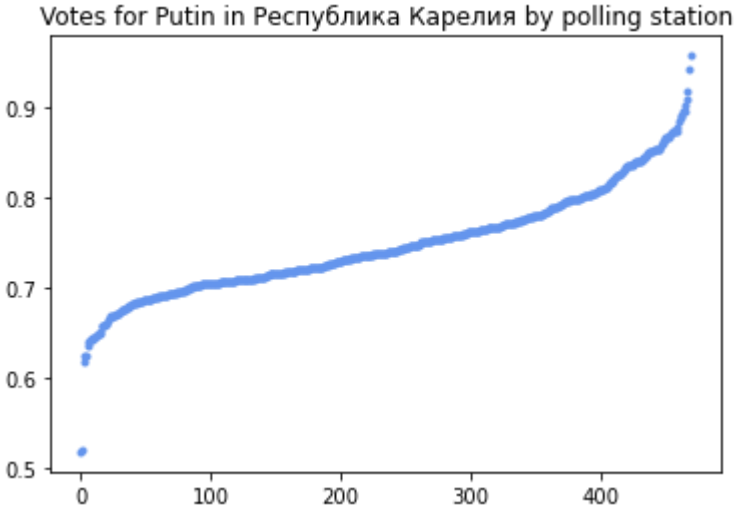
In [31]:
```
for i in range(1, 6):
    visualize_region_results(df_regions_sorted['REGION'][df_regions.shap
```

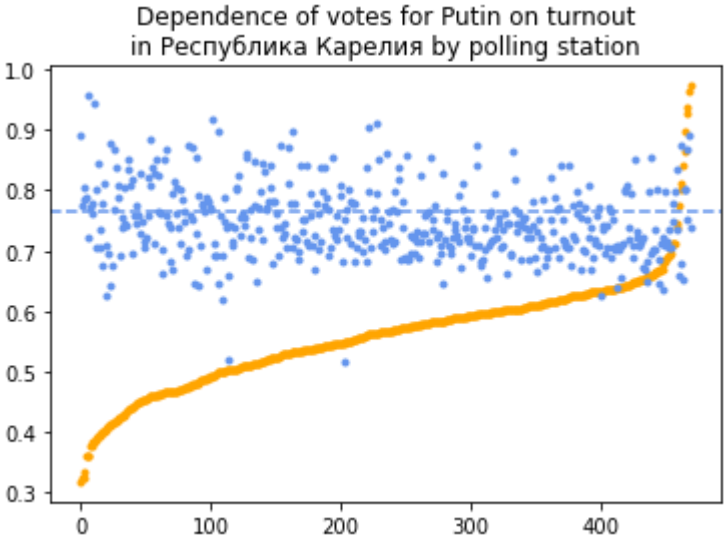


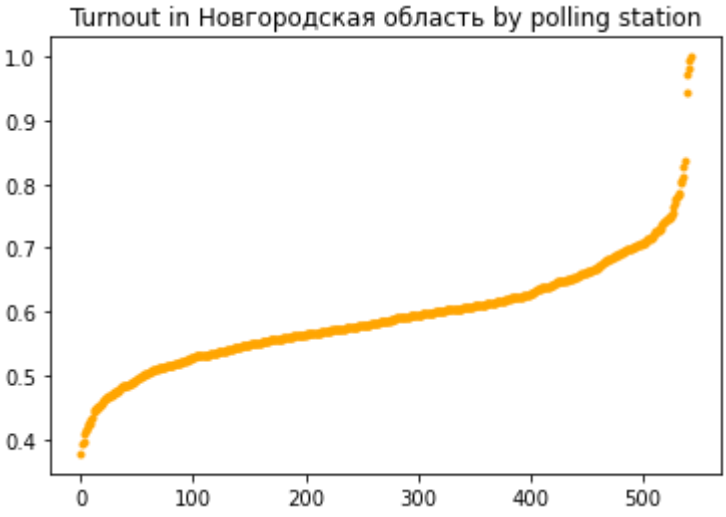

Votes for Putin in Иркутская область by polling station
1.0
0.9
0.8
0.7
0.6
0.5
0
250
500
750
1000
1250
1500
1750
2000


Dependence of votes for Putin on turnout
in Иркутская область by polling station
1.0
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0
250
500
750
1000
1250
1500
1750
2000

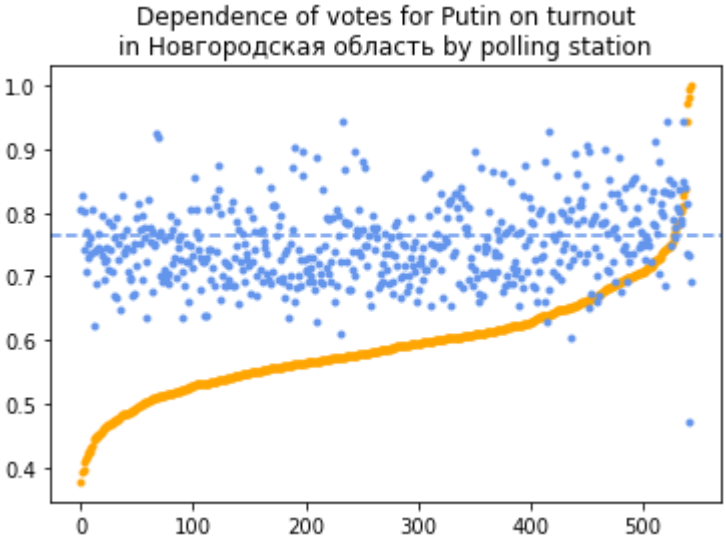

Turnout in Республика Карелия by polling station
1.0
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0
100
200
300
400

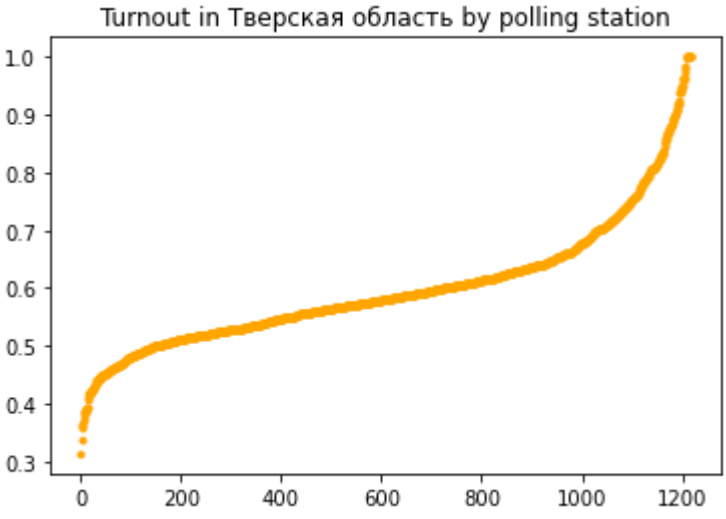

Votes for Putin in Республика Карелия by polling station
0.9
0.8
0.7
0.6
0.5
0
100
200
300
400

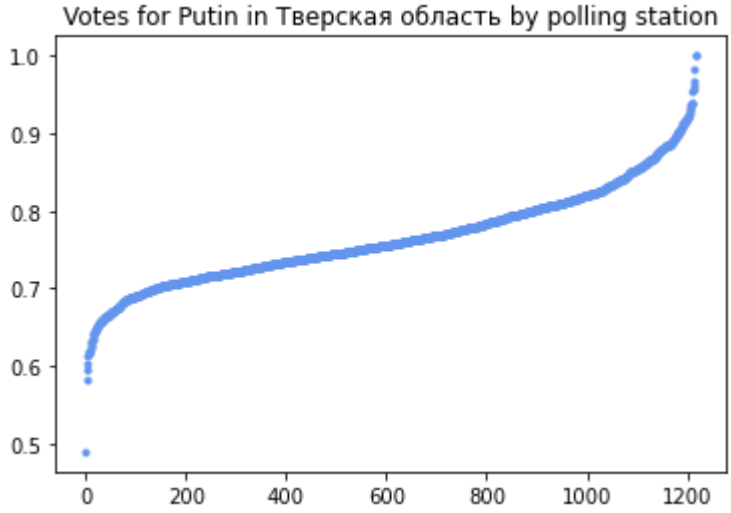

Dependence of votes for Putin on turnout
in Республика Карелия by polling station
1.0
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0
100
200
300
400

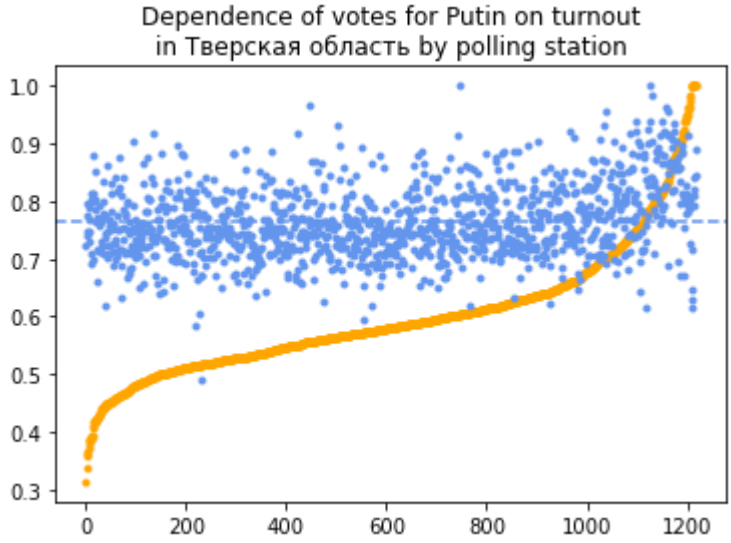

Turnout in Новгородская область by polling station
1.0
0.9
0.8
0.7
0.6
0.5
0.4
0
100
200
300
400
500

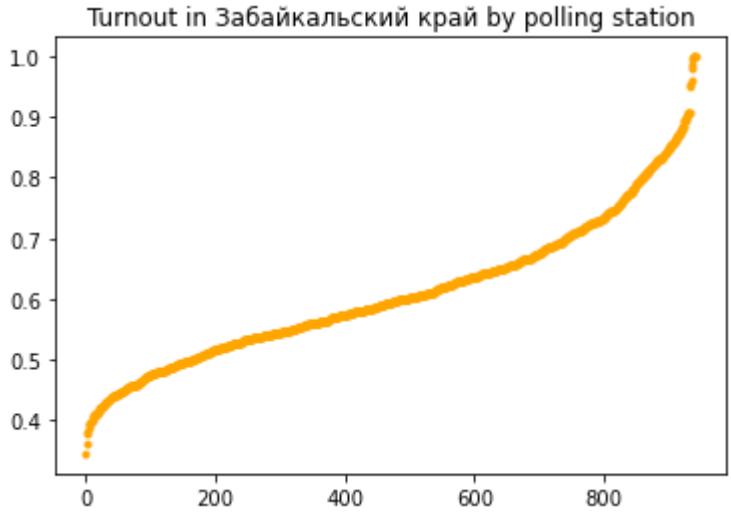

Votes for Putin in Новгородская область by polling station
0.9
0.8
0.7
0.6
0.5
0
100
200
300
400
500

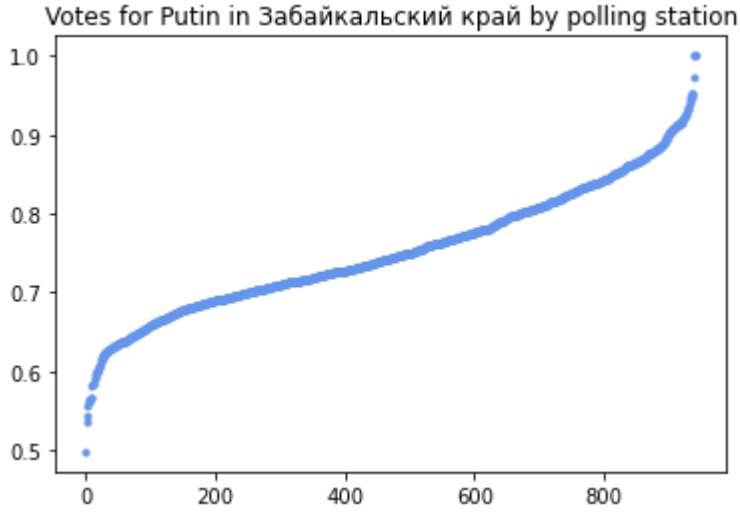

Dependence of votes for Putin on turnout
in Новгородская область by polling station
1.0
0.9
0.8
0.7
0.6
0.5
0.4
0
100
200
300
400
500

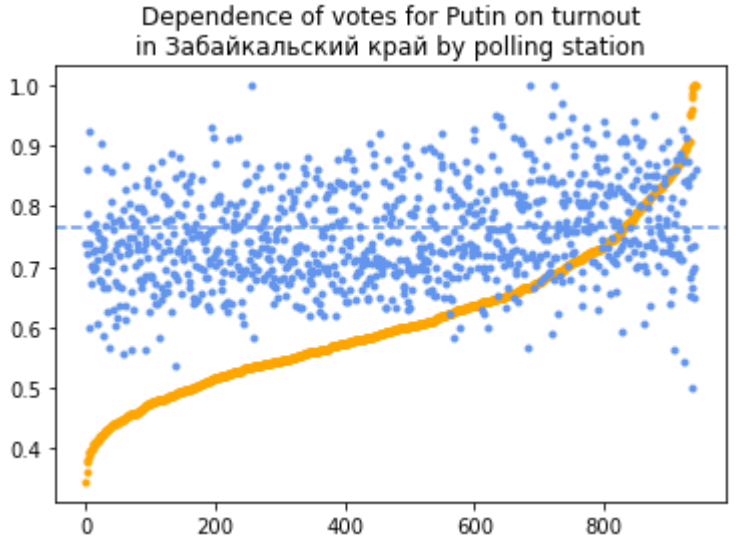

Turnout in Тверская область by polling station
1.0
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0
200
400
600
800
1000
1200


Votes for Putin in Тверская область by polling station
1.0
0.9
0.8
0.7
0.6
0.5
0
200
400
600
800
1000
1200


Dependence of votes for Putin on turnout
in Тверская область by polling station
1.0
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0
200
400
600
800
1000
1200

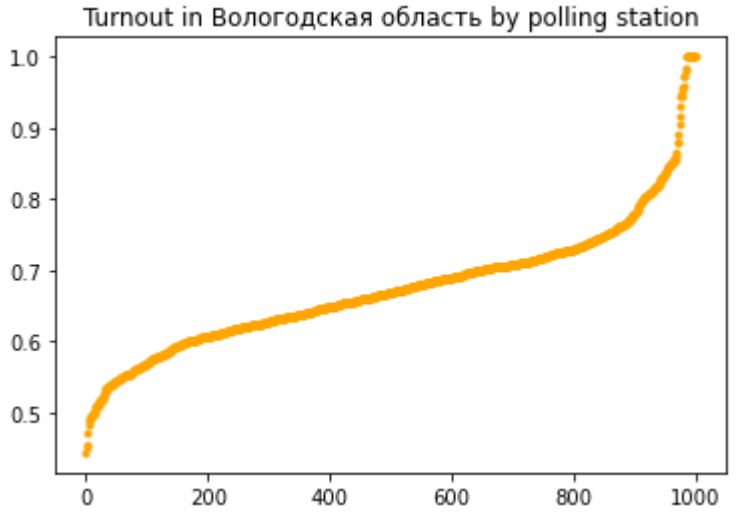

Turnout in Забайкальский край by polling station
1.0
0.9
0.8
0.7
0.6
0.5
0.4
0
200
400
600
800

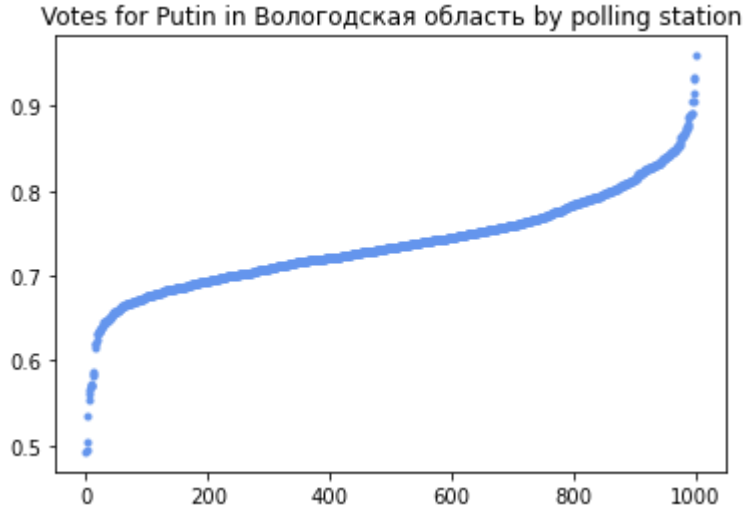


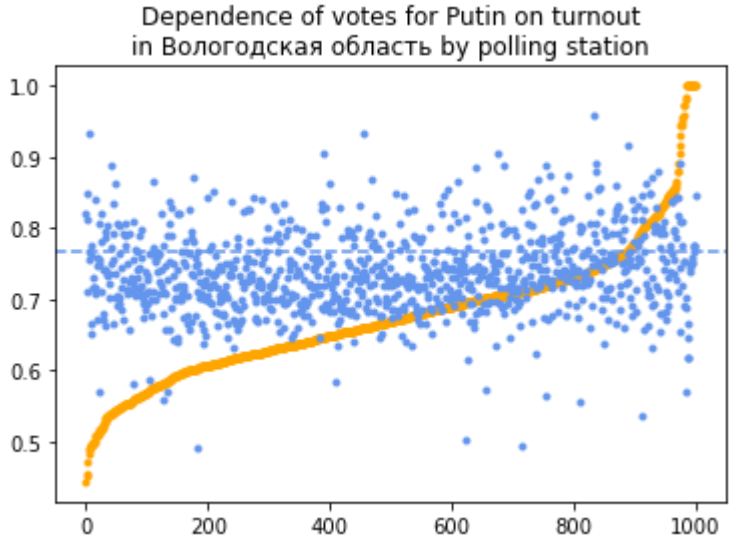


Заметим, что во всех пяти регионах как явка, так и голоса за Путина распределены более-менее равномерно. Голоса расположены по обе стороны от средней линии, независимо от явкиРассмотрим регион из середины списка. Получаем аналогичные результаты

In [32]:
```
visualize_region_results(df_regions['REGION'][40])
```



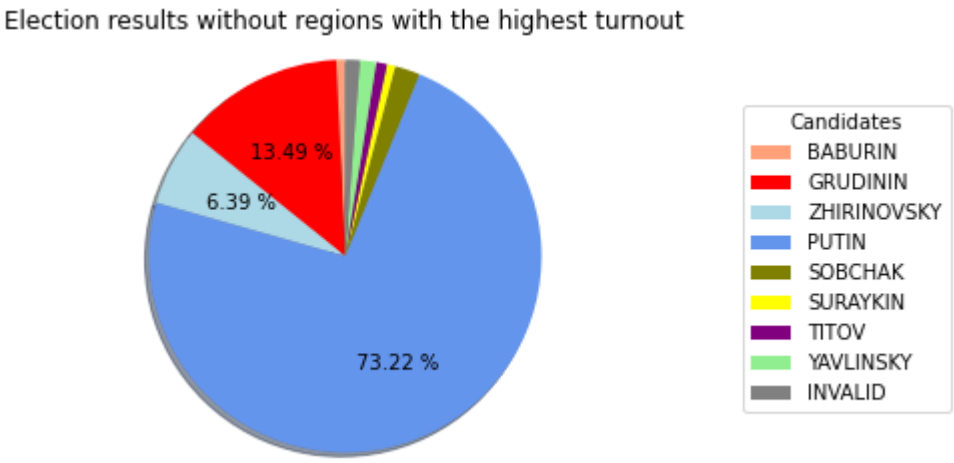




Посмотрим, что будет, если посчитать результаты выборов в той половине регионов, где низкая явка

In [33]:

```
labels = CANDS.copy() + ['INVALID']
start = list(df_regions_sorted.columns).index('BABURIN')
end = list(df_regions_sorted.columns).index('YAVLINSKY') + 1
sizes = [df_regions_sorted[45:][i].sum() for i in df_regions_sorted.colu
sizes.append(df_regions_sorted['N_INVALID'][45:].sum() + df_regions_sort

_, ax = plt.subplots()
ax.pie(sizes, labels=labels,
        autopct=(lambda x: f'{x:.2f} %' if x > 5.0 else ''),
        shadow=True, startangle=90, labeldistance=None,
        colors=COLORS + ['grey'])
ax.axis('equal')
ax.legend(labels, title="Candidates", loc="center", bbox_to_anchor=(1, 0

plt.title('Election results without regions with the highest turnout')
plt.show()
```



Видим, что Путин потерял больше трёх процентов. Хотя, казалось бы, люди должны голосовать примерно равномерно (во всяком случае, распределение голосов может быть обусловлено осособенностями региона, а не явкой)