

**Васильев С. Ю.**

## **Предсказание результатов выборов в Палату Представителей Национального собрания Беларуси в 2016 году**

Набор данных, использованный в данной работе, доступен по ссылке:

[github.com/ushchent/el\\_machina](https://github.com/ushchent/el_machina)

Работа выполнялась на компьютере под управлением Windows 7, с помощью дистрибутива Anaconda 2.4.1 (64-bit), Python 3.5.2.

### **I. Загрузка и подготовка данных**

Импортируем необходимые библиотеки:

```
import pandas as pd
from sklearn import preprocessing
from sklearn.metrics import accuracy_score
from sklearn.cross_validation import train_test_split
from sklearn.svm import SVC
```

Загружаем данные из файла:

```
data = pd.read_csv('Bel_Elec_data.csv')
```

Изучим данные. Исходный набор данных содержит 1985 строк, соответствующих кандидатам на выборах в Палату представителей Национального собрания Беларуси в 2000, 2004, 2008, 2012 и 2016 годах. В наборе 14 столбцов, содержащих разную информацию о кандидатах:

1. region – данные о регионе, в котором расположен избирательный округ (7 значений – 6 областей Беларуси и г. Минск)
2. fio – фамилия, имя и отчество кандидата
3. pol - пол кандидата («м» и «ж»)
4. born – год рождения кандидата (целые числа в диапазоне 1929 – 1994)
5. info – краткая биографическая информация о кандидате. Информация в данном столбце не представлена в едином формате, и её полнота сильно отличается у разных кандидатов, например у одного кандидата просто «пенсионер», а у другого «КОВШ Николай Константинович, 1942 г.р., учитель математики и физики, пенсионер, г.Брест, член ОГП»

6. `deputat` – информация о том, является ли кандидат действующим депутатом Палаты представителей (0 – не является, 1 – является)
7. `party` – информация о партийной принадлежности кандидата
8. `place` – населенный пункт, место жительства кандидата
9. `okrug` – название избирательного округа
10. `n_okrug` – номер избирательного округа
11. `year` – год проведения выборов (2000, 2004, 2008, 2012, 2016)
12. `status` – наш целевой столбец. По выборам 2000 – 2012 содержит информацию, был ли избран кандидат депутатом (0 – не избран, 1 – избран). Для кандидатов на выборах 2016 данный столбец либо не заполнен, либо имеет значение «2» - кандидат зарегистрировался, но сам снялся до выборов.
13. `comment` – содержит информацию о 4 кандидатах, которые были избраны не на основных выборах, а позже, вместо депутатов, досрочно прекративших свои обязанности.
14. `vydvinit` – содержит информацию о способе выдвижения кандидатов на выборах 2016 года (сбор подписей, трудовой коллектив, политическая партия)

Добавим в набор данных столбец `age` – возраст кандидата на момент проведения выборов, в которых он участвует:

```
data['age'] = data['year'] - data['born']
```

Отредактируем данные в столбце `party`:

а) В одной записи имеется опечатка в названии партии, исправим ее:

```
data['party'] =  
data['party'].str.replace('Белорусская социальнр-спортивная  
партия', 'Белорусская социально-спортивная партия')
```

б) Значения «None» и «неизвестно» заменим на «беспартийный»:

```
data['party'] = data['party'].str.replace('None', 'беспартийный')  
data['party'] = data['party'].str.replace('неизвестно', 'беспартийный')
```

Добавим столбец `opporoz`, который будет содержать следующие значения: 0 – кандидат беспартийный, 1 – кандидат представляет

политическую партию, оппозиционную президенту, 2 - кандидат представляет политическую партию, неопозиционную президенту. Информацию об оппозиционности партий возьмем из [Википедии](#).

```
data['oppoz'] = data['party']
vals_to_replace = {'беспартийный': '0',
                   'Белорусская аграрная партия': '2',
                   'Белорусская партия «Зеленые»': '1',
                   'Белорусская партия левых «Справедливый мир»': '1',
                   'Белорусская патриотическая партия': '2',
                   'Белорусская социал-демократическая партия (Грамада)': '1',
                   'Белорусская социал-демократическая партия (Народная
грамада)': '1',
                   'Белорусская социально-спортивная партия': '2',
                   'Коммунистическая партия Беларуси': '2',
                   'Либерально-демократическая партия': '2',
                   'Объединенная гражданская партия': '1',
                   'Партия БНФ': '1',
                   'Партия коммунистов белорусская': '1',
                   'Республиканская партия': '2',
                   'Республиканская партия труда и справедливости': '2',
                   'Социал-демократическая партия народного согласия': '2'
                  }
data['oppoz'] = data['oppoz'].map(vals_to_replace)
```

Разделим столбец fio на три столбца: fam – фамилия, im – имя, otch – отчество.

```
data['fam'], data['im'], data['otch'] =
zip(*data['fio'].map(lambda x: x.split(' ')))
```

Удалим из набора столбцы, которые не будем использовать в машинном обучении

```
del data['fio'], data['okrug'], data['info'], data['vydvinut'], data['comment']
```

Столбец fio удаляем, так как мы разнесли информацию из него по трем столбцам, okrug удаляем, так как у нас есть столбец n\_okrug, который содержит ту же информацию. Столбец info содержит неформатированную

информацию, для использования которой нужна дополнительная обработка. Столбец `yudvinut` содержит информацию только по кандидатам 2016 года. Столбец `comment` содержит информацию только по 4 кандидатам из 1985.

Удаляем из кандидатов, которые снялись до выборов

```
data = data.loc[data.status != 2]
```

## II. Подготовка данных к использованию в алгоритмах машинного обучения

После предварительной подготовки данных мы имеем набор, состоящий из 1957 строк и 14 столбцов.

Index	region	pol	born	deputat	party	place	n_okrug	year	status	age	oppoz	fam	im	otch
0	Брестская область	м	1958	1	беспартийный	г. Минск	11	2004	1	46	0	Кучинский	Виктор	Францевич
1	Брестская область	м	1954	1	беспартийный	г. Минск	14	2004	1	50	0	Андреев	Юрий	Иванович
2	Витебская область	м	1960	1	беспартийный	г. Витебск	18	2004	1	44	0	Семашко	Сергей	Александрович
3	Витебская область	м	1963	1	беспартийный	г. Витебск	19	2004	1	41	0	Иванов	Глеб	Анатолевич
4	Витебская область	м	1957	1	беспартийный	г. Витебск	20	2004	1	47	0	Гудков	Александр	Борисович
5	Витебская область	м	1957	1	беспартийный	г. Минск	22	2004	1	47	0	Хрол	Василий	Петрович
6	Витебская область	м	1950	1	беспартийный	г. Орша	27	2004	1	54	0	Жвиков	Владимир	Владимирович
7	Витебская область	м	1954	1	беспартийный	г. Полоцк	28	2004	1	50	0	Исаев	Владимир	Михайлович
8	Витебская область	м	1953	1	беспартийный	г. Минск	29	2004	1	51	0	Сосонко	Михаил	Павлович
9	Витебская область	м	1947	1	беспартийный	г. Витебск	32	2004	1	57	0	Лекторов	Валерий	Николаевич
10	Гомельская область	м	1945	1	беспартийный	г. Минск	34	2004	1	59	0	Астапченко	Юрий	Леонидович
11	Гомельская область	м	1954	1	беспартийный	г. Гомель	39	2004	1	50	0	Соловьев	Анатолий	Михайлович
12	Гомельская область	м	1941	1	Коммунистиче...	г. Мозырь	45	2004	1	63	2	Костян	Сергей	Иванович
13	Гомельская область	м	1954	1	Белорусская аграрная пар...	г. Речица	46	2004	1	50	2	Куцко	Николай	Васильевич
14	Гомельская область	м	1946	1	беспартийный	г. Рогачев	47	2004	1	58	0	Поплавский	Михаил	Иванович
15	Гомельская область	м	1958	1	Белорусская аграрная пар...	г. Речица	49	2004	1	46	2	Кулик	Виталий	Васильевич
16	Гродненская область	м	1961	1	беспартийный	г. Гродно	51	2004	1	43	0	Каменецкий	Сергей	Владимирович

Выполним `LabelEncoding`, т. е. обозначим все показатели цифрами

```
data_encoded = data.apply(preprocessing.LabelEncoder().fit_transform)
```

Получаем набор следующего вида:



Index	region	pol	born	deputat	party	place	n_okrug	year	status	age	oppoz	fam	im	otch
0	0	1	26	1	15	184	10	1	1	25	0	694	30	182
1	0	1	22	1	15	184	13	1	1	29	0	35	132	84
2	1	1	28	1	15	149	17	1	1	23	0	1130	112	6
3	1	1	31	1	15	149	18	1	1	20	0	473	44	11
4	1	1	25	1	15	149	19	1	1	26	0	292	2	32
5	1	1	25	1	15	184	21	1	1	26	0	1328	28	142
6	1	1	18	1	15	196	26	1	1	33	0	403	35	54
7	1	1	22	1	15	201	27	1	1	29	0	485	35	123
8	1	1	21	1	15	184	28	1	1	30	0	1185	91	138
9	1	1	15	1	15	149	31	1	1	36	0	726	26	131
10	2	1	13	1	15	96	33	1	1	38	0	64	132	108
11	2	1	22	1	15	66	38	1	1	29	0	1181	10	123
12	2	1	9	1	7	99	44	1	1	42	2	625	112	84
13	2	1	22	1	0	114	45	1	1	29	2	691	96	41

Выполним нормализацию данных во всех столбцах, кроме status и year

```
col_for_scaling = list(data.columns)
del col_for_scaling[8], col_for_scaling[7]
for i in col_for_scaling:
    data_encoded[i] = preprocessing.scale(data_encoded[i])
```

Получаем набор нормализованных данных

Index	region	pol	born	deputat	party	place	n_okrug	year	status	age	oppoz	fam	im	otch
0	-1.52	0.472	-0.259	3.23	0.869	0.534	-1.38	1	1	-0.0422	-0.919	-0.0849	-0.709	1.83
1	-1.52	0.472	-0.586	3.23	0.869	0.534	-1.29	1	1	0.347	-0.919	-1.65	1.82	-0.0369
2	-1.01	0.472	-0.0961	3.23	0.869	0.0217	-1.17	1	1	-0.237	-0.919	0.947	1.33	-1.52
3	-1.01	0.472	0.149	3.23	0.869	0.0217	-1.14	1	1	-0.529	-0.919	-0.608	-0.361	-1.42
4	-1.01	0.472	-0.341	3.23	0.869	0.0217	-1.11	1	1	0.0551	-0.919	-1.04	-1.4	-1.03
5	-1.01	0.472	-0.341	3.23	0.869	0.534	-1.05	1	1	0.0551	-0.919	1.42	-0.758	1.07
6	-1.01	0.472	-0.912	3.23	0.869	0.709	-0.902	1	1	0.736	-0.919	-0.774	-0.585	-0.607
7	-1.01	0.472	-0.586	3.23	0.869	0.782	-0.872	1	1	0.347	-0.919	-0.58	-0.585	0.705
8	-1.01	0.472	-0.667	3.23	0.869	0.534	-0.842	1	1	0.444	-0.919	1.08	0.805	0.99
9	-1.01	0.472	-1.16	3.23	0.869	0.0217	-0.752	1	1	1.03	-0.919	-0.009...	-0.808	0.857
10	-0.506	0.472	-1.32	3.23	0.869	-0.754	-0.692	1	1	1.22	-0.919	-1.58	1.82	0.419
11	-0.506	0.472	-0.586	3.23	0.869	-1.19	-0.541	1	1	0.347	-0.919	1.07	-1.2	0.705
12	-0.506	0.472	-1.65	3.23	-1.01	-0.71	-0.361	1	1	1.61	1.47	-0.248	1.33	-0.0369
13	-0.506	0.472	-0.586	3.23	-2.65	-0.49	-0.331	1	1	0.347	1.47	-0.092	0.929	-0.855

Разделим данные на два набора: data00\_12 - кандидаты на выборах 2000 – 2012, и data16 – кандидаты на выборах 2016.

```
data00_12 = data_encoded.loc[data_encoded.year < 5]
```

```
data16 = data_encoded.loc[data_encoded.year > 5]
```

Удалим неиспользуемые столбцы

```
del data00_12['year'], data16['year'], data16['status']
```

Определим список столбцов-признаков

```
features = list(data00_12.columns)
del features[7]
```

Разделим data00\_12 на тестовую и тренировочную выборки

```
X_train, X_test, y_train, y_test = train_test_split(data00_12[features],
                                                    data00_12.status,
                                                    test_size=0.20,
                                                    random_state=123)
```

### III. Обучение алгоритма

Попробуем выполнить обучение алгоритма C-Support Vector Classification с параметрами по умолчанию

```
svc = SVC(probability = True, random_state = 42)
svc = svc.fit(X_train, y_train)
y_pred = svc.predict(X_test)
```

Посмотрим точность предсказания

```
accuracy_score(y_test, y_pred)
```

Точность составила - 0.777397260274

Для использования модели на данных 2016 года, переобучим ее на всех данных 2000 – 2012

```
svc_full = SVC(probability = True, random_state = 42)
svc_full = svc_full.fit(data00_12[features], data00_12['status'])
```

Выполним предсказание для данных 2016 года. При этом необходимо учитывать следующий момент. Если использовать метод `.predict()`, то мы получим ответы 0 (для кандидата с вероятностью победы менее 0,5) либо 1 (для кандидата с вероятностью победы более 0,5) для каждого кандидата. В

результате на некоторых участках у нас получится более одного «победителя», а на некоторых - ни одного. Но на каждом участке должен быть один победитель.

Поэтому, для определения победителя, мы используем метод `.predict_proba()`, который в качестве ответа даст нам вероятность победы каждого кандидата. Затем мы выберем на каждом участке кандидата с наибольшей вероятностью победы и будем считать его победителем.

```
res_svc = svc_full.predict_proba(data16)
```

#### IV. Оформление итоговых результатов

Создадим итоговый набор данных с результатами

```
data2016 = data.loc[data.year > 2015]
data2016 = data2016[['n_okrug', 'fam', 'im', 'otch']]
data2016['real'] = pd.Series(result2016, index=data2016.index)
data2016['prob'] = pd.Series(res_svc[:,1], index=data2016.index)
```

Здесь `result2016` – список с итогами выборов 2016.

```
ok_res = pd.DataFrame(columns=['n_okrug', 'fam', 'im', 'otch', 'real', 'prob'])
for i in list(set(data2016['n_okrug'])):
    okr = data2016.loc[data2016['n_okrug'] == i]
    max_okr = max(okr['prob'])
    okr1 = okr.loc[okr['prob'] == max_okr]
    ok_res = ok_res.append(okr1)
```

В итоге наш алгоритм правильно предсказал результаты голосования на 86 участках из 110, то есть точность составила 0,78

Код проекта приведен в файле – `Bel_Parl_election.py`

Результаты предсказания в файле – `Bel_Elec_Pred.pdf`

Официальные итоги выборов в файле – `Bel_Elec_Of_Itog.pdf`