

Защищено:
Гапанюк Ю.Е.

Демонстрация:
Гапанюк Ю.Е.

"__" _____ 2022 г.

"__" _____ 2022 г.

**Отчет по рубежному контролю № 2 по курсу
Технологии машинного обучения
ГУИМЦ**

**Тема работы: " Методы построения моделей машинного
обучения. "**

16
(количество листов)
Вариант № 28

ИСПОЛНИТЕЛЬ:

студент группы ИУ5Ц-82Б

Чиварзин А.Е.

(подпись)

"__" _____ 2022 г.

Цель лабораторной работы

Изучение линейных моделей, SVM и деревьев решений

Задание

1. Выберите набор данных (датасет) для решения задачи классификации или регрессии.
2. В случае необходимости проведите удаление или заполнение пропусков и кодирование категориальных признаков.
3. С использованием метода `train_test_split` разделите выборку на обучающую и тестовую.
4. Обучите следующие модели:
 - одну из линейных моделей (линейную или полиномиальную регрессию при решении задачи регрессии, логистическую регрессию при решении задачи классификации);
 - SVM;
 - дерево решений.
5. Оцените качество моделей с помощью двух подходящих для задачи метрик. Сравните качество полученных моделей.
6. Постройте график, показывающий важность признаков в дереве решений.
7. Визуализируйте дерево решений или выведите правила дерева решений в текстовом виде.

Ход лабораторной работы

Текстовое описание набора данных

В качестве набора данных используется dataset рейтингов университетов мира на основании трёх рейтингов. Датасет доступен по адресу: <https://www.kaggle.com/mylesoneill/world-university-rankings>

Из набора данных будет рассматриваться только файл `cwurData.csv`.

Описание столбцов:

- `world_rank` - мировой рейтинг университета
- `institution` - название университета
- `country` - страна, в которой расположен университет
- `national_rank` - рейтинг университета в стране его нахождения
- `quality_of_education` - рейтинг качества образования
- `quality_of_faculty` - рейтинг качества профессорско-преподавательского состава
- `publications` - рейтинг публикаций
- `influence` - рейтинг влияния
- `citations` - количество студентов в университете
- `broad_impact` - рейтинг за широкое влияние (предоставлен только за 2014 и 2015 гг. Остальное - пропуски)
- `patents` - рейтинг за патенты
- `score` - общий балл, используемый для определения мирового рейтинга
- `year` - год рейтинга (с 2012 по 2015 год)

Основные характеристики набора данных

Подключаем все необходимые библиотеки

```
In [1]: import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib
import matplotlib_inline
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
%matplotlib inline
sns.set(style="ticks")
from io import StringIO
```

Подключаем Dataset

```
In [2]: data = pd.read_csv('cwurData.csv', sep=",")
```

Размер набора данных

```
In [3]: data.shape
```

```
Out[3]: (2200, 14)
```

Типы колонок

```
In [4]: data.dtypes
```

```
Out[4]: world_rank      int64
        institution     object
        country         object
        national_rank    int64
        quality_of_education int64
        alumni_employment int64
        quality_of_faculty int64
        publications     int64
        influence        int64
        citations        int64
        broad_impact     float64
        patents          int64
        score            float64
        year             int64
        dtype: object
```

Проверяем, есть ли пропущенные значения

```
In [5]: data.isnull().sum()
```

```
Out[5]: world_rank      0
        institution     0
        country         0
        national_rank    0
        quality_of_education 0
        alumni_employment 0
        quality_of_faculty 0
        publications     0
        influence        0
        citations        0
        broad_impact     200
        patents          0
        score            0
        year             0
        dtype: int64
```

Первые 5 строк датасета

```
In [6]: data.head()
```

```
Out[6]:
```

	world_rank	institution	country	national_rank	quality_of_education	alumni_employment	quality_of_faculty	publications	influence
0	1	Harvard University	USA	1	7	9	1	1	1
1	2	Massachusetts Institute of Technology	USA	2	9	17	3	12	4
2	3	Stanford University	USA	3	17	11	5	4	2
3	4	University of Cambridge	United Kingdom	1	10	24	4	16	16
4	5	California Institute of Technology	USA	4	2	29	7	37	22

```
In [7]: total_count = data.shape[0]
        print('Всего строк: {}'.format(total_count))
```

Всего строк: 2200

Процент пропусков в broad_impact

```
In [8]: (200 / 2200) * 100
```

```
Out[8]: 9.090909090909092
```

Настройка отображения графиков

```
In [9]: # Задание формата графиков для сохранения высокого качества PNG
        from IPython.display import set_matplotlib_formats
        matplotlib_inline.backend_inline.set_matplotlib_formats("retina")
        # Задание ширины графиков, чтобы они помещались на A4
        pd.set_option("display.width", 70)
```

Обработка пропусков данных

Очистка строк

Можно очистить строки, содержащие пропуски. При этом останутся данные только за 2014 и 2015 гг (см. описание датасета)

```
In [10]: # Удаление строк, содержащих пустые значения
data_no_null = data.dropna(axis=0, how='any')
(data.shape, data_no_null.shape)
```

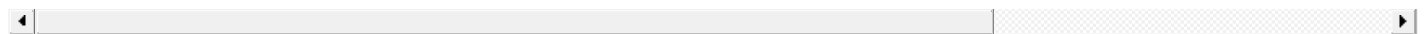
```
Out[10]: ((2200, 14), (2000, 14))
```

Выведем первые 11 строк, чтобы убедиться, что данные в `national_rank` числовые (Jupyter Lab в предпросмотре CSV показывает не совсем верно)

```
In [11]: data_no_null.head(11)
```

```
Out[11]:
```

	world_rank	institution	country	national_rank	quality_of_education	alumni_employment	quality_of_faculty	publications	influence
200	1	Harvard University	USA	1	1	1	1	1	
201	2	Stanford University	USA	2	11	2	4	5	
202	3	Massachusetts Institute of Technology	USA	3	3	11	2	15	
203	4	University of Cambridge	United Kingdom	1	2	10	5	10	
204	5	University of Oxford	United Kingdom	2	7	12	10	11	
205	6	Columbia University	USA	4	13	8	9	14	
206	7	University of California, Berkeley	USA	5	4	22	6	7	
207	8	University of Chicago	USA	6	10	14	8	17	
208	9	Princeton University	USA	7	5	16	3	70	
209	10	Yale University	USA	8	9	25	11	18	
210	11	Cornell University	USA	9	12	18	19	23	



```
In [12]: total_count = data_no_null.shape[0]
print('Всего строк: {}'.format(total_count))
```

Всего строк: 2000

Кодирование категориальных признаков

Преобразуем названия стран, городов, ... в числовые значения (label encoding)

```
In [13]: from sklearn.preprocessing import LabelEncoder, OneHotEncoder
```

```
In [14]: le = LabelEncoder()
# "institution"
le.fit(data_no_null.institution.drop_duplicates())
data_no_null.institution = le.transform(data_no_null.institution)
# "country"
le.fit(data_no_null["country"].drop_duplicates())
data_no_null["country"] = le.transform(data_no_null["country"])
```

```
/tmp/ipykernel_143/4210865855.py:4: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead
```

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy

```
data_no_null.institution = le.transform(data_no_null.institution)
/tmp/ipykernel_143/4210865855.py:7: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead
```

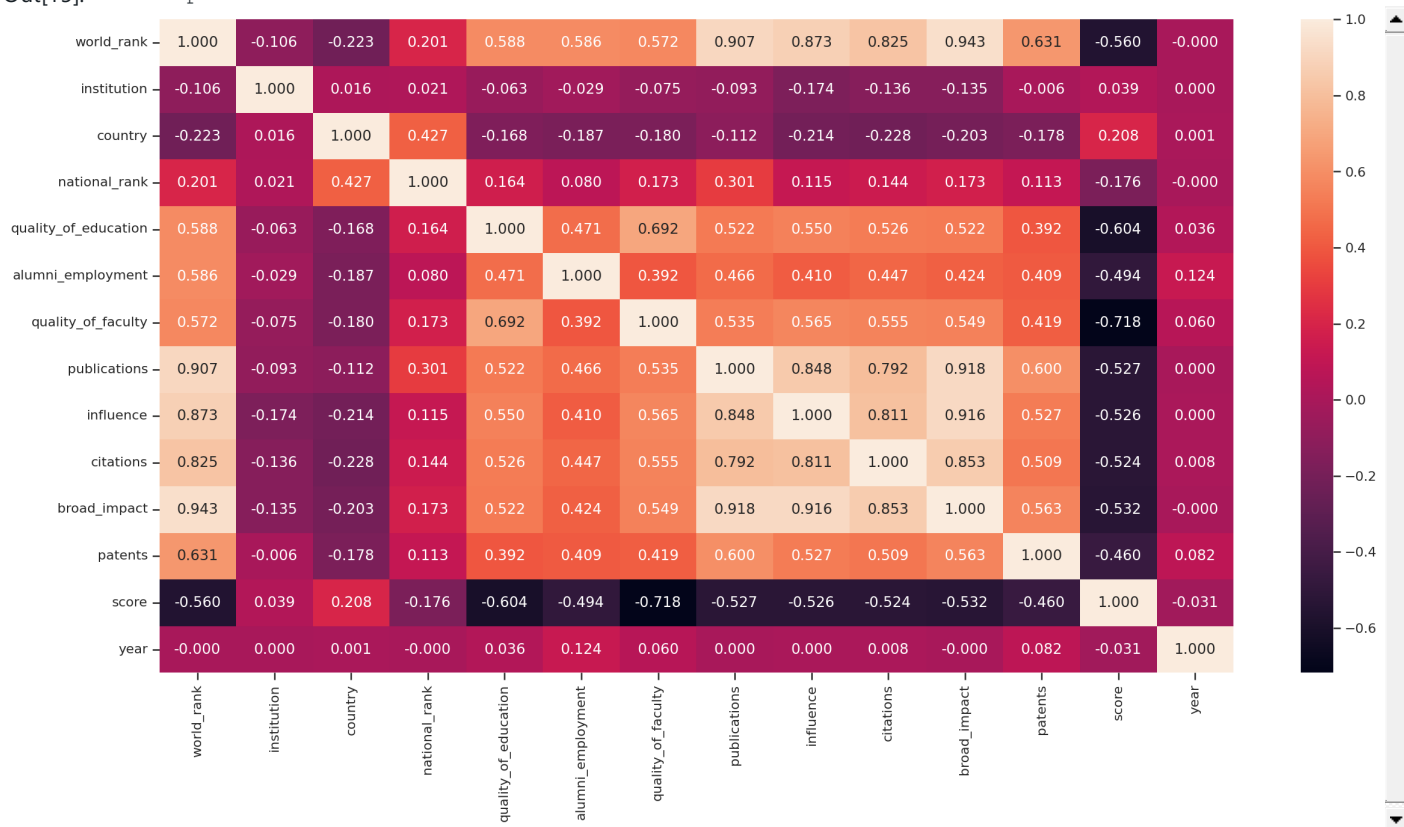
See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy

```
data_no_null["country"] = le.transform(data_no_null["country"])
```

Построим корреляционную матрицу

```
In [15]: ig, ax = plt.subplots(figsize=(20,10))
sns.heatmap(data_no_null.corr(method='pearson'), ax=ax, annot=True, fmt='.3f')
```

Out[15]:<AxesSubplot:>



Предсказание целевого признака

Предскажем значение целевого признака `world_rank` по `broad_impact` и `publications`, поскольку их значения корреляции ближе всего к 1

Разбиение выборки на обучающую и тестовую

```
In [16]: X = data_no_null[["broad_impact", "publications"]]
Y = data_no_null["world_rank"]
print('Входные данные:\n\n', X.head(), '\n\nВыходные данные:\n\n', Y.head())
```

Входные данные:

	broad_impact	publications
200	1.0	1
201	4.0	5
202	2.0	15
203	13.0	10
204	12.0	11

Выходные данные:

200	1
201	2
202	3
203	4
204	5

Name: world_rank, dtype: int64

Разделим выборку на обучающую и тестовую

```
In [17]: X_train, X_test, Y_train, Y_test = train_test_split(X, Y, random_state = 2022, test_size = 0.1)
```

Входные параметры обучающей выборки

```
In [18]: X_train.head()
```

```
Out[18]:
```

	broad_impact	publications
2164	932.0	875
1710	590.0	576
428	164.0	200
1389	164.0	233
2089	932.0	675

Входные параметры тестовой выборки

```
In [19]: X_test.head()
```

```
Out[19]:
```

	broad_impact	publications
1218	14.0	3
1495	265.0	236
843	703.0	943
2042	850.0	803
1869	606.0	701

Выходные параметры обучающей выборки

```
In [20]: Y_train.head()
```

```
Out[20]:
```

2164	965
1710	511
428	229
1389	190
2089	890

Name: world_rank, dtype: int64

Выходные параметры тестовой выборки

```
In [21]: Y_test.head()
```

```
Out[21]:
```

1218	19
1495	296
843	644
2042	843
1869	670

Name: world_rank, dtype: int64

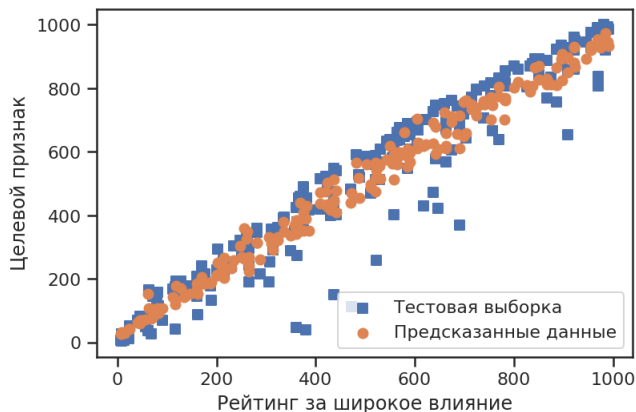
Построение линейной регрессии

```
In [22]: from sklearn.linear_model import LinearRegression
         from sklearn.metrics import mean_absolute_error, mean_squared_error, median_absolute_error, r2_score
```

```
In [23]: Lin_Reg = LinearRegression().fit(X_train, Y_train)
         lr_y_pred = Lin_Reg.predict(X_test)
```

Возьмем тот параметр, чья корреляция ближе всего к единице, т.е. broad_impact

```
In [40]: plt.scatter(X_test["broad_impact"], Y_test, marker = 's', label = 'Тестовая выборка')
plt.scatter(X_test["broad_impact"], lr_y_pred, marker = 'o', label = 'Предсказанные данные')
plt.legend (loc = 'lower right')
plt.xlabel ('Рейтинг за широкое влияние')
plt.ylabel ('Целевой признак')
plt.show()
```



```
In [25]: from sklearn.metrics import mean_absolute_error, mean_squared_error, median_absolute_error, r2_score
```

```
In [26]: print('Средняя абсолютная ошибка:', mean_absolute_error(Y_test, lr_y_pred))
print('Средняя квадратичная ошибка:', mean_squared_error(Y_test, lr_y_pred))
print('Median absolute error:', median_absolute_error(Y_test, lr_y_pred))
print('Коэффициент детерминации:', r2_score(Y_test, lr_y_pred))
```

Средняя абсолютная ошибка: 54.70203008487861

Средняя квадратичная ошибка: 6228.270901286782

Median absolute error: 42.05433711920929

Коэффициент детерминации: 0.9234584275958889

SVM

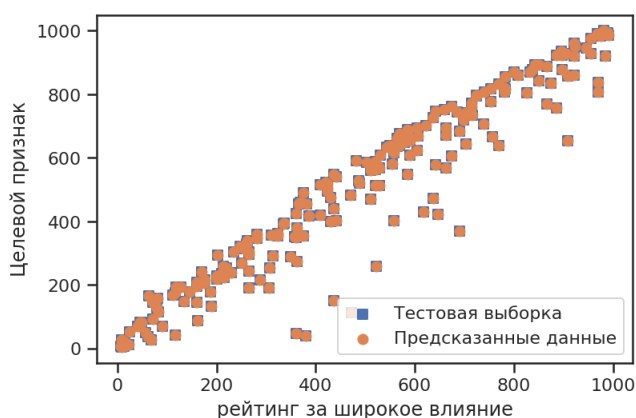
```
In [27]: from sklearn.svm import SVC , LinearSVC
from sklearn.datasets import make_blobs
```

```
In [28]: svc = SVC(kernel='linear')
svc.fit(X_train,Y_train)
```

```
Out[28]: SVC(kernel='linear')
```

```
In [29]: pred_y = svc.predict(X_test)
```

```
In [41]: plt.scatter(X_test["broad_impact"], Y_test, marker = 's', label = 'Тестовая выборка')
plt.scatter(X_test["broad_impact"], pred_y, marker = 'o', label = 'Предсказанные данные')
plt.legend (loc = 'lower right')
plt.xlabel ('рейтинг за широкое влияние')
plt.ylabel ('Целевой признак')
plt.show()
```



```
In [31]: print('Средняя абсолютная ошибка:', mean_absolute_error(Y_test, pred_y))
print('Средняя квадратичная ошибка:', mean_squared_error(Y_test, pred_y))
print('Median absolute error:', median_absolute_error(Y_test, pred_y))
print('Коэффициент детерминации:', r2_score(Y_test, pred_y))
```

Средняя абсолютная ошибка: 57.19

Средняя квадратичная ошибка: 9379.4

Median absolute error: 27.0

Коэффициент детерминации: 0.8847330124868531

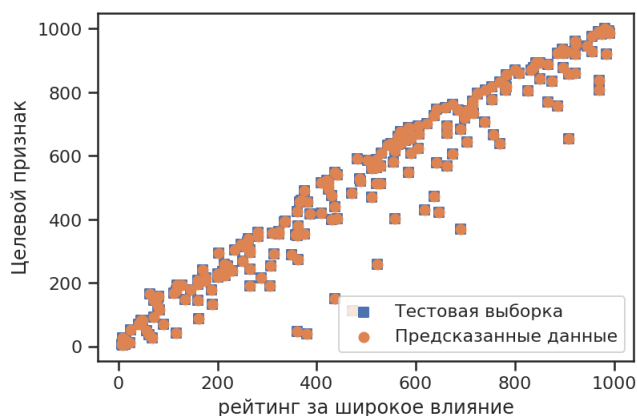
Дерево (Tree)

```
In [32]: from sklearn.tree import DecisionTreeClassifier, DecisionTreeRegressor, export_graphviz
from sklearn.tree import export_graphviz
from sklearn import tree
import re
from IPython.core.display import HTML
from sklearn.tree import export_text
import graphviz
from IPython.display import Image
import pydotplus
```

Обучим дерево на всех признаках

```
In [33]: reg = tree.DecisionTreeRegressor()  
         reg = reg.fit(X test, Y test)
```

```
In [42]: pred_y = reg.predict(X_test)
plt.scatter(X_test["broad_impact"], Y_test, marker = 's', label = 'Тестовая выборка')
plt.scatter(X_test["broad_impact"], pred_y, marker = 'o', label = 'Предсказанные данные')
plt.legend(loc = 'lower right')
plt.xlabel ('рейтинг за широкое влияние')
plt.ylabel ('Целевой признак')
plt.show()
```



Дерево в текстовом виде

```
In [35]: tree_rules = export_text(reg, feature_names=list(X.columns))
         HTML('<pre>' + tree_rules + '</pre>')
```

[illegible]

[illegible]

[illegible]

```
| | | | |-- value: [353.00]
| | | | |--- publications > 382.00
| | | | |   |-- value: [40.00]
| | | | |--- publications > 399.00
| | | | |   |-- broad_impact <= 348.00
| | | | |     |-- broad_impact <= 298.50
| | | | |       |-- broad_impact <= 258.00
| | | | |         |-- broad_impact <= 251.50
| | | | |           |-- value: [321.00]
| | | | |             |-- broad_impact > 251.50
| | | | |               |-- value: [317.00]
| | | | |                 |-- broad_impact > 258.00
| | | | |                   |-- publications <= 516.00
| | | | |                     |-- value: [329.00]
| | | | |                       |-- publications > 516.00
| | | | |                         |-- value: [340.00]
| | | | |                           |-- broad_impact > 298.50
| | | | |                             |-- value: [392.00]
| | | | |                               |-- broad_impact > 348.00
| | | | |                                 |-- broad_impact <= 363.50
| | | | |                                   |-- value: [425.00]
| | | | |                                     |-- broad_impact > 363.50
| | | | |                                       |-- broad_impact <= 372.00
| | | | |                                         |-- publications <= 509.50
| | | | |                                           |-- value: [456.00]
| | | | |                                             |-- publications > 509.50
| | | | |                                               |-- value: [461.00]
| | | | |                                                 |-- broad_impact > 372.00
| | | | |                                                   |-- value: [491.00]
| | | | |--- broad_impact > 380.50
| | | | |   |-- broad_impact <= 471.50
| | | | |     |-- publications <= 551.00
| | | | |       |-- publications <= 455.50
| | | | |         |-- publications <= 321.00
| | | | |           |-- publications <= 265.00
| | | | |             |-- value: [418.00]
| | | | |               |-- publications > 265.00
| | | | |                 |-- value: [402.00]
| | | | |                   |-- publications > 321.00
| | | | |                     |-- broad_impact <= 427.00
| | | | |                       |-- publications <= 436.00
| | | | |                         |-- value: [495.00]
| | | | |                           |-- publications > 436.00
| | | | |                             |-- value: [505.00]
| | | | |                               |-- broad_impact > 427.00
| | | | |                                 |-- publications <= 386.00
| | | | |                                   |-- publications <= 351.50
| | | | |                                     |-- value: [484.00]
| | | | |                                       |-- publications > 351.50
| | | | |                                         |-- value: [474.00]
| | | | |                                           |-- publications > 386.00
| | | | |                                             |-- broad_impact <= 433.50
| | | | |                                               |-- value: [400.00]
| | | | |                                                 |-- broad_impact > 433.50
| | | | |                                                   |-- value: [441.00]
| | | | |--- publications > 455.50
| | | | |   |-- publications <= 477.50
| | | | |     |-- value: [150.00]
| | | | |       |-- publications > 477.50
| | | | |         |-- publications <= 518.50
| | | | |           |-- value: [419.00]
| | | | |             |-- publications > 518.50
| | | | |               |-- value: [454.00]
| | | | |--- publications > 551.00
| | | | |   |-- broad_impact <= 430.50
| | | | |     |-- publications <= 621.00
| | | | |       |-- value: [524.00]
| | | | |         |-- publications > 621.00
| | | | |           |-- broad_impact <= 416.50
| | | | |             |-- value: [516.00]
| | | | |               |-- broad_impact > 416.50
| | | | |                 |-- value: [519.00]
| | | | |--- broad_impact > 430.50
| | | | |   |-- broad_impact <= 439.50
| | | | |     |-- value: [547.00]
| | | | |       |-- broad_impact > 439.50
| | | | |         |-- value: [541.00]
```

```

| | | |--- broad_impact > 471.50
| | | |--- value: [114.00]
|--- broad_impact > 477.00
| |--- broad_impact <= 709.00
| | |--- publications <= 626.50
| | | |--- broad_impact <= 694.50
| | | |--- publications <= 592.50
| | | |--- publications <= 458.00
| | | |--- broad_impact <= 539.00
| | | |--- publications <= 434.50
| | | |--- broad_impact <= 499.50
| | | |--- value: [520.00]
| | | |--- broad_impact > 499.50
| | | |--- broad_impact <= 515.50
| | | |--- value: [562.00]
| | | |--- broad_impact > 515.50
| | | |--- value: [563.00]
| | | |--- publications > 434.50
| | | |--- value: [469.00]
| | | |--- broad_impact > 539.00
| | | |--- publications <= 390.50
| | | |--- value: [401.00]
| | | |--- publications > 390.50
| | | |--- value: [370.00]
| | | |--- publications > 458.00
| | | |--- publications <= 510.00
| | | |--- publications <= 505.50
| | | |--- broad_impact <= 652.50
| | | |--- broad_impact <= 558.00
| | | |--- broad_impact <= 525.50
| | | |--- value: [589.00]
| | | |--- broad_impact > 525.50
| | | |--- value: [608.00]
| | | |--- broad_impact > 558.00
| | | |--- publications <= 463.50
| | | |--- value: [548.00]
| | | |--- publications > 463.50
| | | |--- value: [579.00]
| | | |--- broad_impact > 652.50
| | | |--- publications <= 488.00
| | | |--- value: [672.00]
| | | |--- publications > 488.00
| | | |--- value: [568.00]
| | | |--- publications > 505.50
| | | |--- value: [422.00]
| | | |--- publications > 510.00
| | | |--- broad_impact <= 638.00
| | | |--- broad_impact <= 559.50
| | | |--- value: [581.00]
| | | |--- broad_impact > 559.50
| | | |--- publications <= 567.50
| | | |--- publications <= 528.50
| | | |--- value: [654.00]
| | | |--- publications > 528.50
| | | |--- value: [637.00]
| | | |--- publications > 567.50
| | | |--- value: [678.00]
| | | |--- broad_impact > 638.00
| | | |--- value: [745.00]
| | | |--- publications > 592.50
| | | |--- publications <= 596.50
| | | |--- broad_impact <= 504.50
| | | |--- value: [529.00]
| | | |--- broad_impact > 504.50
| | | |--- value: [258.00]
| | | |--- publications > 596.50
| | | |--- broad_impact <= 573.50
| | | |--- value: [568.00]
| | | |--- broad_impact > 573.50
| | | |--- broad_impact <= 627.50
| | | |--- value: [429.00]
| | | |--- broad_impact > 627.50
| | | |--- value: [473.00]
| | | |--- broad_impact > 694.50
| | | |--- broad_impact <= 701.00
| | | |--- value: [729.00]
| | | |--- broad_impact > 701.00

```

```
| | | | |--- value: [741.00]
| | | | |--- publications > 626.50
| | | | |--- broad_impact <= 536.00
| | | | |--- publications <= 699.00
| | | | |--- publications <= 664.00
| | | | |--- value: [512.00]
| | | | |--- publications > 664.00
| | | | |--- value: [513.00]
| | | | |--- publications > 699.00
| | | | |--- publications <= 754.50
| | | | |--- value: [585.00]
| | | | |--- publications > 754.50
| | | | |--- value: [590.00]
| | | | |--- broad_impact > 536.00
| | | | |--- broad_impact <= 614.00
| | | | |--- publications <= 628.00
| | | | |--- value: [608.00]
| | | | |--- publications > 628.00
| | | | |--- broad_impact <= 561.50
| | | | |--- broad_impact <= 553.50
| | | | |--- publications <= 740.00
| | | | |--- value: [633.00]
| | | | |--- publications > 740.00
| | | | |--- value: [639.00]
| | | | |--- broad_impact > 553.50
| | | | |--- value: [616.00]
| | | | |--- broad_impact > 561.50
| | | | |--- publications <= 943.00
| | | | |--- broad_impact <= 582.00
| | | | |--- publications <= 662.50
| | | | |--- value: [638.00]
| | | | |--- publications > 662.50
| | | | |--- publications <= 686.50
| | | | |--- value: [676.00]
| | | | |--- publications > 686.50
| | | | |--- truncated branch of depth 2
| | | | |--- broad_impact > 582.00
| | | | |--- publications <= 630.50
| | | | |--- value: [660.00]
| | | | |--- publications > 630.50
| | | | |--- broad_impact <= 605.50
| | | | |--- truncated branch of depth 2
| | | | |--- broad_impact > 605.50
| | | | |--- value: [670.00]
| | | | |--- publications > 943.00
| | | | |--- value: [623.00]
| | | | |--- broad_impact > 614.00
| | | | |--- broad_impact <= 661.00
| | | | |--- publications <= 715.00
| | | | |--- value: [701.00]
| | | | |--- publications > 715.00
| | | | |--- broad_impact <= 639.50
| | | | |--- value: [728.00]
| | | | |--- broad_impact > 639.50
| | | | |--- publications <= 856.00
| | | | |--- value: [748.00]
| | | | |--- publications > 856.00
| | | | |--- value: [752.00]
| | | | |--- broad_impact > 661.00
| | | | |--- publications <= 780.00
| | | | |--- broad_impact <= 669.00
| | | | |--- value: [693.00]
| | | | |--- broad_impact > 669.00
| | | | |--- value: [762.00]
| | | | |--- publications > 780.00
| | | | |--- broad_impact <= 682.50
| | | | |--- value: [607.00]
| | | | |--- broad_impact > 682.50
| | | | |--- publications <= 938.50
| | | | |--- publications <= 867.00
| | | | |--- value: [684.00]
| | | | |--- publications > 867.00
| | | | |--- value: [719.00]
| | | | |--- publications > 938.50
| | | | |--- value: [644.00]
| | | | |--- broad_impact > 709.00
| | | | |--- broad_impact <= 915.00
```

```

| | | |--- broad_impact <= 769.50
| | | |--- broad_impact <= 755.00
| | | |--- broad_impact <= 740.00
| | | |--- publications <= 803.50
| | | |--- publications <= 753.50
| | | |--- value: [797.00]
| | | |--- publications > 753.50
| | | |--- value: [795.00]
| | | |--- publications > 803.50
| | | |--- broad_impact <= 715.50
| | | |--- publications <= 838.50
| | | |--- value: [773.00]
| | | |--- publications > 838.50
| | | |--- value: [759.00]
| | | |--- broad_impact > 715.50
| | | |--- broad_impact <= 727.50
| | | |--- value: [735.00]
| | | |--- broad_impact > 727.50
| | | |--- value: [707.00]
| | | |--- broad_impact > 740.00
| | | |--- publications <= 686.50
| | | |--- value: [776.00]
| | | |--- publications > 686.50
| | | |--- broad_impact <= 747.50
| | | |--- value: [808.00]
| | | |--- broad_impact > 747.50
| | | |--- value: [818.00]
| | | |--- broad_impact > 755.00
| | | |--- broad_impact <= 762.50
| | | |--- value: [667.00]
| | | |--- broad_impact > 762.50
| | | |--- value: [639.00]
| | | |--- broad_impact > 769.50
| | | |--- publications <= 824.00
| | | |--- broad_impact <= 903.00
| | | |--- publications <= 651.00
| | | |--- broad_impact <= 833.50
| | | |--- value: [807.00]
| | | |--- broad_impact > 833.50
| | | |--- value: [756.00]
| | | |--- publications > 651.00
| | | |--- publications <= 760.50
| | | |--- broad_impact <= 807.50
| | | |--- publications <= 750.00
| | | |--- publications <= 732.50
| | | |--- value: [818.00]
| | | |--- publications > 732.50
| | | |--- value: [819.00]
| | | |--- publications > 750.00
| | | |--- value: [855.00]
| | | |--- broad_impact > 807.50
| | | |--- broad_impact <= 870.50
| | | |--- publications <= 699.00
| | | |--- value: [889.00]
| | | |--- publications > 699.00
| | | |--- truncated branch of depth 2
| | | |--- broad_impact > 870.50
| | | |--- publications <= 670.50
| | | |--- value: [878.00]
| | | |--- publications > 670.50
| | | |--- value: [834.00]
| | | |--- publications > 760.50
| | | |--- publications <= 799.50
| | | |--- broad_impact <= 846.50
| | | |--- value: [804.00]
| | | |--- broad_impact > 846.50
| | | |--- value: [770.00]
| | | |--- publications > 799.50
| | | |--- publications <= 806.50
| | | |--- value: [843.00]
| | | |--- publications > 806.50
| | | |--- value: [833.00]
| | | |--- broad_impact > 903.00
| | | |--- value: [653.00]
| | | |--- publications > 824.00
| | | |--- broad_impact <= 868.00
| | | |--- publications <= 919.50

```

```
| | | | | | |--- broad_impact <= 804.50  
| | | | | | |--- value: [871.00]  
| | | | | | |--- broad_impact > 804.50  
| | | | | | |--- value: [861.00]  
| | | | | | |--- publications > 919.50  
| | | | | | |--- value: [892.00]  
| | | | | | |--- broad_impact > 868.00  
| | | | | | |--- broad_impact <= 907.50  
| | | | | | |--- broad_impact <= 891.00  
| | | | | | |--- value: [923.00]  
| | | | | | |--- broad_impact > 891.00  
| | | | | | |--- broad_impact <= 896.50  
| | | | | | |--- value: [935.00]  
| | | | | | |--- broad_impact > 896.50  
| | | | | | |--- publications <= 946.50  
| | | | | | |--- value: [929.00]  
| | | | | | |--- publications > 946.50  
| | | | | | |--- value: [924.00]  
| | | | | | |--- broad_impact > 907.50  
| | | | | | |--- value: [858.00]  
| | | | | |--- broad_impact > 915.00  
| | | | | | |--- broad_impact <= 972.50  
| | | | | | |--- broad_impact <= 969.50  
| | | | | | |--- broad_impact <= 922.00  
| | | | | | |--- publications <= 993.50  
| | | | | | |--- publications <= 898.00  
| | | | | | |--- value: [920.00]  
| | | | | | |--- publications > 898.00  
| | | | | | |--- value: [860.00]  
| | | | | | |--- publications > 993.50  
| | | | | | |--- value: [962.00]  
| | | | | | |--- broad_impact > 922.00  
| | | | | | |--- publications <= 894.00  
| | | | | | |--- broad_impact <= 951.00  
| | | | | | |--- broad_impact <= 934.50  
| | | | | | |--- value: [950.00]  
| | | | | | |--- broad_impact > 934.50  
| | | | | | |--- value: [946.00]  
| | | | | | |--- broad_impact > 951.00  
| | | | | | |--- value: [928.00]  
| | | | | | |--- publications > 894.00  
| | | | | | |--- broad_impact <= 962.50  
| | | | | | |--- value: [975.00]  
| | | | | | |--- broad_impact > 962.50  
| | | | | | |--- value: [992.00]  
| | | | | | |--- broad_impact > 969.50  
| | | | | | |--- publications <= 936.00  
| | | | | | |--- value: [837.00]  
| | | | | | |--- publications > 936.00  
| | | | | | |--- value: [807.00]  
| | | | | | |--- broad_impact > 972.50  
| | | | | | |--- publications <= 945.00  
| | | | | | |--- broad_impact <= 978.00  
| | | | | | |--- publications <= 873.50  
| | | | | | |--- value: [986.00]  
| | | | | | |--- publications > 873.50  
| | | | | | |--- value: [983.00]  
| | | | | | |--- broad_impact > 978.00  
| | | | | | |--- publications <= 840.00  
| | | | | | |--- value: [985.00]  
| | | | | | |--- publications > 840.00  
| | | | | | |--- broad_impact <= 985.00  
| | | | | | |--- value: [1000.00]  
| | | | | | |--- broad_impact > 985.00  
| | | | | | |--- value: [994.00]  
| | | | | | |--- publications > 945.00  
| | | | | | |--- value: [920.00]
```

Визуализация дерева

```
In [36]: # Визуализация дерева
```

```
def get_png_tree(tree_model_param, feature_names_param):
    dot_data = StringIO()
    export_graphviz(tree_model_param, out_file=dot_data, feature_names=feature_names_param,
                    filled=True, rounded=True, special_characters=True)
    graph = pydotplus.graph_from_dot_data(dot_data.getvalue())
```

```
return graph.create_png()
```

```
In [37]: Image(get_png_tree(reg, X.columns), height='100%')
```

Out[37]:

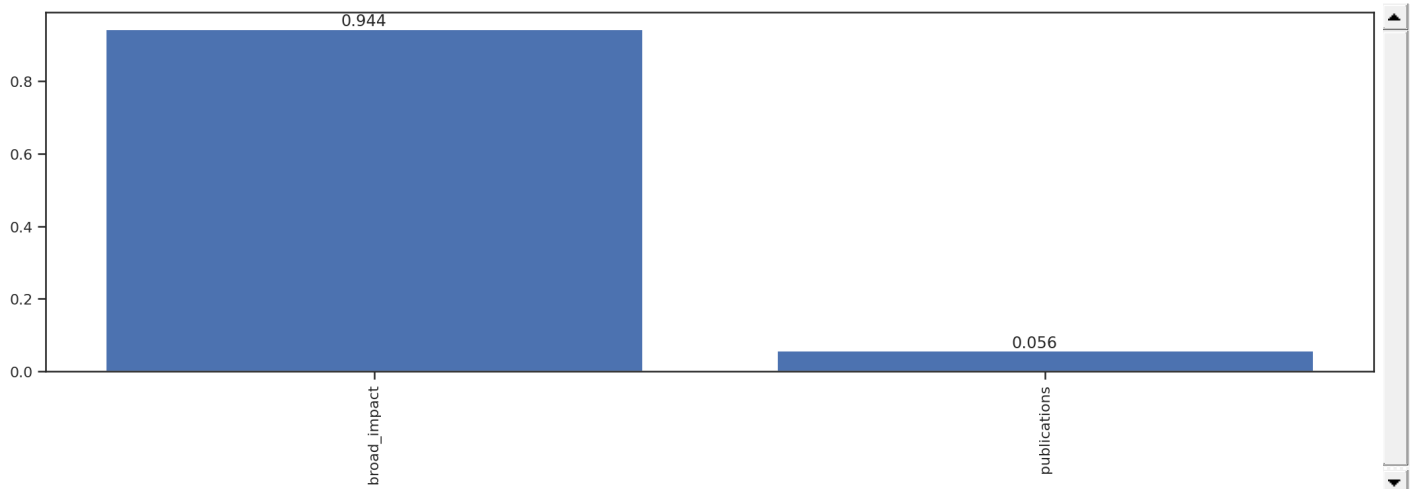


Важность признаков в дереве

```
In [38]: from operator import itemgetter
```

```
def draw_feature_importances(tree_model, X_dataset, figsize=(18,5)):
    """
    Вывод важности признаков в виде графика
    """
    # Сортировка значений важности признаков по убыванию
    list_to_sort = list(zip(X_dataset.columns.values, tree_model.feature_importances_))
    sorted_list = sorted(list_to_sort, key=itemgetter(1), reverse = True)
    # Названия признаков
    labels = [x for x, _ in sorted_list]
    # Важности признаков
    data = [x for _, x in sorted_list]
    # Вывод графика
    fig, ax = plt.subplots(figsize=figsize)
    ind = np.arange(len(labels))
    plt.bar(ind, data)
    plt.xticks(ind, labels, rotation='vertical')
    # Вывод значений
    for a,b in zip(ind, data):
        plt.text(a-0.05, b+0.01, str(round(b,3)))
    plt.show()
    return labels, data
```

```
In [39]: boston_tree_regr_fl, boston_tree_regr_fd = draw_feature_importances(reg, X)
```



In []: