Московский государственный технический университет имени Н. Э. Баумана

Кафедра «Системы обработки информации и управления»

Рубежный контроль №1
по курсу
«Методы машинного обучения»

Выполнил:

Студент ИУ5-24М

Черната Н. С.

Москва, 2020

Задание

Для заданного набора данных проведите обработку пропусков в данных. Какие способы обработки пропусков в данных для категориальных и количественных признаков Вы использовали? Какие признаки Вы будете использовать для дальнейшего построения моделей машинного обучения и почему? Для заданного набора данных произведите масштабирование данных и преобразование категориальных признаков в количественные. Какие методы Вы использовали для решения задачи и почему?

Набор данных:

https://www.kaggle.com/karangadiya/fifa19

In [1]:
```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
%matplotlib inline
sns.set(style="ticks")
```

Загрузка данных

In [3]:
```
data = pd.read_csv('MMO/fifa19/data.csv', sep=",")
```

In [74]:
```
data.shape
```

Out[74]:
```
(18207, 89)
```

In [5]:
```
data.dtypes
```

Out[5]:
```
Unnamed: 0          int64
ID                  int64
Name               object
Age                 int64
Photo              object
                    ...
GKHandling        float64
GKKicking         float64
GKPositioning     float64
GKReflexes        float64
Release Clause     object
Length: 89, dtype: object
```

In [6]:
```
data.isnull().sum()
```

Out[6]:
```
Unnamed: 0           0
ID                   0
Name                 0
Age                  0
```

In [7]:
```
data.head()
```

Out[7]:

	Unnamed: 0	ID	Name	Age	Photo	Nationality	Flag	Overall	Potential	Club	...	Co
0	0	158023	L. Messi	31	https://cdn.sofifa.org/players/4/19/158023.png	Argentina	https://cdn.sofifa.org/flags/52.png	94	94	FC Barcelona	...	
1	1	20801	Cristiano Ronaldo	33	https://cdn.sofifa.org/players/4/19/20801.png	Portugal	https://cdn.sofifa.org/flags/38.png	94	94	Juventus	...	
2	2	190871	Neymar Jr	26	https://cdn.sofifa.org/players/4/19/190871.png	Brazil	https://cdn.sofifa.org/flags/54.png	92	93	Paris Saint-Germain	...	
3	3	193080	De Gea	27	https://cdn.sofifa.org/players/4/19/193080.png	Spain	https://cdn.sofifa.org/flags/45.png	91	93	Manchester United	...	
4	4	192985	K. De Bruyne	27	https://cdn.sofifa.org/players/4/19/192985.png	Belgium	https://cdn.sofifa.org/flags/7.png	91	92	Manchester City	...	

5 rows × 89 columns

In [8]:
```
total_count = data.shape[0]
print('Всего строк: {}'.format(total_count))
```

```
Всего строк: 18207
```

Обработка пропусков в данных

Удаление или заполнение нулями

In [9]:
```
data_new_1 = data.dropna(axis=1, how='any')
(data.shape, data_new_1.shape)
```

Out[9]: ((18207, 89), (18207, 13))

In [10]:
```
data_new_2 = data.dropna(axis=0, how='any')
(data.shape, data_new_2.shape)
```

Out[10]: ((18207, 89), (0, 89))

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

Out[11]:

	Unnamed: 0	ID	Name	Age	Photo	Nationality	Flag	Overall	Potential	Club	...	Co
0	0	158023	L. Messi	31	https://cdn.sofifa.org/players/4/19/158023.png	Argentina	https://cdn.sofifa.org/flags/52.png	94	94	FC Barcelona	...	
1	1	20801	Cristiano Ronaldo	33	https://cdn.sofifa.org/players/4/19/20801.png	Portugal	https://cdn.sofifa.org/flags/38.png	94	94	Juventus	...	
2	2	190871	Neymar Jr	26	https://cdn.sofifa.org/players/4/19/190871.png	Brazil	https://cdn.sofifa.org/flags/54.png	92	93	Paris Saint-Germain	...	
3	3	193080	De Gea	27	https://cdn.sofifa.org/players/4/19/193080.png	Spain	https://cdn.sofifa.org/flags/45.png	91	93	Manchester United	...	
4	4	192985	K. De Bruyne	27	https://cdn.sofifa.org/players/4/19/192985.png	Belgium	https://cdn.sofifa.org/flags/7.png	91	92	Manchester City	...	

5 rows × 89 columns

Импьютация

Обработка пропусков в числовых данных

In [12]:
```
data_new_3 = data.fillna(0)
data_new_3.head()
```

Out[12]:

	Unnamed: 0	ID	Name	Age	Photo	Nationality	Flag	Overall	Potential	Club	...	Co
0	0	158023	L. Messi	31	https://cdn.sofifa.org/players/4/19/158023.png	Argentina	https://cdn.sofifa.org/flags/52.png	94	94	FC Barcelona	...	
1	1	20801	Cristiano Ronaldo	33	https://cdn.sofifa.org/players/4/19/20801.png	Portugal	https://cdn.sofifa.org/flags/38.png	94	94	Juventus	...	
2	2	190871	Neymar Jr	26	https://cdn.sofifa.org/players/4/19/190871.png	Brazil	https://cdn.sofifa.org/flags/54.png	92	93	Paris Saint-Germain	...	
3	3	193080	De Gea	27	https://cdn.sofifa.org/players/4/19/193080.png	Spain	https://cdn.sofifa.org/flags/45.png	91	93	Manchester United	...	
4	4	192985	K. De Bruyne	27	https://cdn.sofifa.org/players/4/19/192985.png	Belgium	https://cdn.sofifa.org/flags/7.png	91	92	Manchester City	...	

5 rows × 89 columns

In [13]:
```
num_cols = []
for col in data.columns:
    # Количество пустых значений
    temp_null_count = data[data[col].isnull()].shape[0]
    dt = str(data[col].dtype)
    if temp_null_count>0 and (dt=='float64' or dt=='int64'):
        num_cols.append(col)
        temp_perc = round((temp_null_count / total_count) * 100.0, 2)
        print('Колонка {}. Тип данных {}. Количество пустых значений {}, {}%.'.format(col, dt, temp_null_count, temp_perc))
```

```
Колонка International Reputation. Тип данных float64. Количество пустых значений 48, 0.26%.
Колонка Weak Foot. Тип данных float64. Количество пустых значений 48, 0.26%.
Колонка Skill Moves. Тип данных float64. Количество пустых значений 48, 0.26%.
```

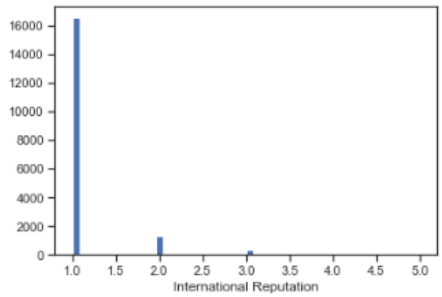
In [14]:
```
data_num = data[num_cols]
data_num
```

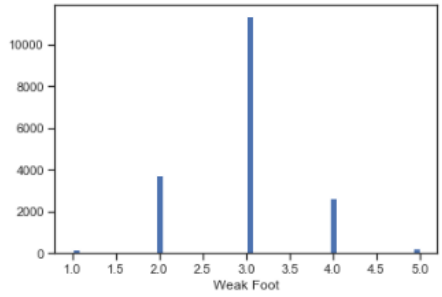
Out[14]:

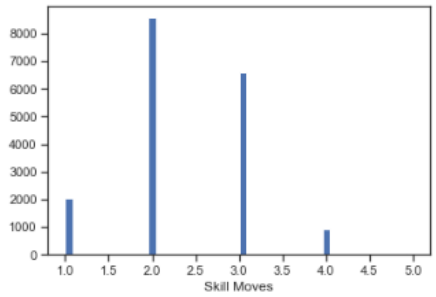
	International Reputation	Weak Foot	Skill Moves	Jersey Number	Crossing	Finishing	HeadingAccuracy	ShortPassing	Volleys	Dribbling	...	Penalties	Composure	Marking	Star
0	5.0	4.0	4.0	10.0	84.0	95.0	70.0	90.0	86.0	97.0	...	75.0	96.0	33.0	
1	5.0	4.0	5.0	7.0	84.0	94.0	89.0	81.0	87.0	88.0	...	85.0	95.0	28.0	
2	5.0	5.0	5.0	10.0	79.0	87.0	62.0	84.0	84.0	96.0	...	81.0	94.0	27.0	
3	4.0	3.0	1.0	1.0	17.0	13.0	21.0	50.0	13.0	18.0	...	40.0	68.0	15.0	
4	4.0	5.0	4.0	7.0	93.0	82.0	55.0	92.0	82.0	86.0	...	79.0	88.0	68.0	
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
18202	1.0	2.0	2.0	22.0	34.0	38.0	40.0	49.0	25.0	42.0	...	43.0	45.0	40.0	
18203	1.0	2.0	2.0	21.0	23.0	52.0	52.0	43.0	36.0	39.0	...	43.0	42.0	22.0	
18204	1.0	3.0	2.0	33.0	25.0	40.0	46.0	38.0	38.0	45.0	...	55.0	41.0	32.0	
18205	1.0	3.0	2.0	34.0	44.0	50.0	39.0	42.0	40.0	51.0	...	50.0	46.0	20.0	
18206	1.0	3.0	2.0	33.0	41.0	34.0	46.0	48.0	30.0	43.0	...	33.0	43.0	40.0	

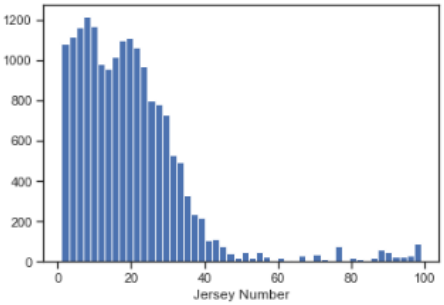
18207 rows × 38 columns

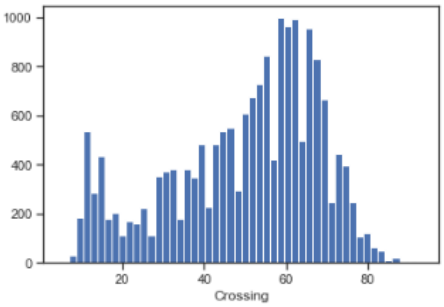
In [15]:
```
for col in data_num:
    plt.hist(data[col], 50)
    plt.xlabel(col)
    plt.show()
```

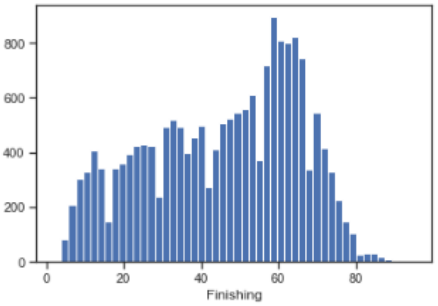

16000
14000
12000
10000
8000
6000
4000
2000
0
1.0 1.5 2.0 2.5 3.0 3.5 4.0 4.5 5.0
International Reputation

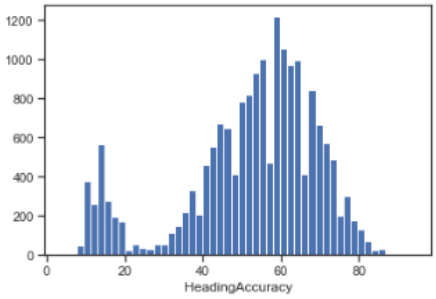

10000
8000
6000
4000
2000
0
1.0 1.5 2.0 2.5 3.0 3.5 4.0 4.5 5.0
Weak Foot

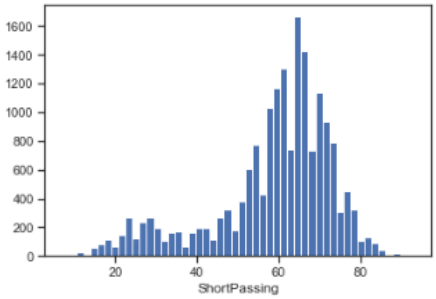

8000
7000
6000
5000
4000
3000
2000
1000
0
1.0 1.5 2.0 2.5 3.0 3.5 4.0 4.5 5.0
Skill Moves

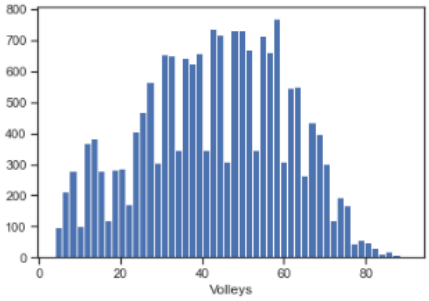

1200
1000
800
600
400
200
0
0 20 40 60 80 100
Jersey Number


1000
800
600
400
200
0
20 40 60 80
Crossing


800
600
400
200
0
0 20 40 60 80
Finishing


1200
1000
800
600
400
200
0
0
20
40
60
80
HeadingAccuracy


1600
1400
1200
1000
800
600
400
200
0
20
40
60
80
ShortPassing


800
700
600
500
400
300
200
100
0
0
20
40
60
80
Volleys

In [18]:
```
data[data['Jersey Number'].isnull()]
```

Out[18]:

	Unnamed: 0	ID	Name	Age	Photo	Nationality	Flag	Overall	Potential	
5018	5018	153160	R. Raldes	37	https://cdn.sofifa.org/players/4/19/153160.png	Bolivia	https://cdn.sofifa.org/flags/53.png	70	70	
6736	6736	175393	J. Arce	33	https://cdn.sofifa.org/players/4/19/175393.png	Bolivia	https://cdn.sofifa.org/flags/53.png	68	68	
7922	7922	195905	L. Gutiérrez	33	https://cdn.sofifa.org/players/4/19/195905.png	Bolivia	https://cdn.sofifa.org/flags/53.png	67	67	
9905	9905	226044	R. Vargas	23	https://cdn.sofifa.org/players/4/19/226044.png	Bolivia	https://cdn.sofifa.org/flags/53.png	66	69	
10628	10628	216751	D. Bejarano	26	https://cdn.sofifa.org/players/4/19/216751.png	Bolivia	https://cdn.sofifa.org/flags/53.png	65	66	
13236	13236	177971	J. McNulty	33	https://cdn.sofifa.org/players/4/19/177971.png	Scotland	https://cdn.sofifa.org/flags/42.png	62	62	Ro
13237	13237	195380	J. Barrera	29	https://cdn.sofifa.org/players/4/19/195380.png	Nicaragua	https://cdn.sofifa.org/flags/86.png	62	62	E Ch
13238	13238	139317	J. Stead	35	https://cdn.sofifa.org/players/4/19/139317.png	England	https://cdn.sofifa.org/flags/14.png	62	62	
13239	13239	240437	A. Semprini	20	https://cdn.sofifa.org/players/4/19/240437.png	Italy	https://cdn.sofifa.org/flags/27.png	62	72	E

In [19]:
```
flt_index = data[data['Jersey Number'].isnull()].index
flt_index
```

Out[19]:
```
Int64Index([ 5018,  6736,  7922,  9905, 10628, 13236, 13237, 13238, 13239,
            13240, 13241, 13242, 13243, 13244, 13245, 13246, 13247, 13248,
            13249, 13250, 13251, 13252, 13253, 13254, 13255, 13256, 13257,
            13258, 13259, 13260, 13261, 13262, 13263, 13264, 13265, 13266,
            13267, 13268, 13269, 13270, 13271, 13272, 13273, 13274, 13275,
            13276, 13277, 13278, 13279, 13280, 13281, 13282, 13283, 16450,
            16539, 16793, 17129, 17339, 17436, 17539],
           dtype='int64')
```

In [20]:
```
data[data.index.isin(flt_index)]
```

Out[20]:

	Unnamed: 0	ID	Name	Age	Photo	Nationality	Flag	Overall	Potential	
5018	5018	153160	R. Raldes	37	https://cdn.sofifa.org/players/4/19/153160.png	Bolivia	https://cdn.sofifa.org/flags/53.png	70	70	
6736	6736	175393	J. Arce	33	https://cdn.sofifa.org/players/4/19/175393.png	Bolivia	https://cdn.sofifa.org/flags/53.png	68	68	
7922	7922	195905	L. Gutiérrez	33	https://cdn.sofifa.org/players/4/19/195905.png	Bolivia	https://cdn.sofifa.org/flags/53.png	67	67	
9905	9905	226044	R. Vargas	23	https://cdn.sofifa.org/players/4/19/226044.png	Bolivia	https://cdn.sofifa.org/flags/53.png	66	69	
10628	10628	216751	D. Bejarano	26	https://cdn.sofifa.org/players/4/19/216751.png	Bolivia	https://cdn.sofifa.org/flags/53.png	65	66	
13236	13236	177971	J. McNulty	33	https://cdn.sofifa.org/players/4/19/177971.png	Scotland	https://cdn.sofifa.org/flags/42.png	62	62	Ro
13237	13237	195380	J. Barrera	29	https://cdn.sofifa.org/players/4/19/195380.png	Nicaragua	https://cdn.sofifa.org/flags/86.png	62	62	E Ch
13238	13238	139317	J. Stead	35	https://cdn.sofifa.org/players/4/19/139317.png	England	https://cdn.sofifa.org/flags/14.png	62	62	
13239	13239	240437	A. Semprini	20	https://cdn.sofifa.org/players/4/19/240437.png	Italy	https://cdn.sofifa.org/flags/27.png	62	72	E

In [21]:
```
data_num[data_num.index.isin(flt_index)]['Jersey Number']
```

Out[21]:
```
5018    NaN
6736    NaN
7922    NaN
9905    NaN
10628   NaN
13236   NaN
13237   NaN
13238   NaN
13239   NaN
13240   NaN
13241   NaN
13242   NaN
13243   NaN
13244   NaN
13245   NaN
13246   NaN
13247   NaN
13248   NaN
13249   NaN
13250   NaN
```

In [22]:
```
data_num_Jersey_Number = data_num[['Jersey Number']]
data_num_Jersey_Number.head()
```

Out[22]:

	Jersey Number
0	10.0
1	7.0
2	10.0
3	1.0
4	7.0

In [24]:
```
from sklearn.impute import SimpleImputer
from sklearn.impute import MissingIndicator
```

In [25]:
```
indicator = MissingIndicator()
mask_missing_values_only = indicator.fit_transform(data_num_Jersey_Number)
mask_missing_values_only
```

Out[25]:
```
array([[False],
       [False],
       [False],
       ...,
       [False],
       [False],
       [False]])
```

In [26]:
```
strategies=['mean', 'median','most_frequent']
```

In [31]:
```
def test_num_impute(strategy_param):
    imp_num = SimpleImputer(strategy=strategy_param)
    data_num_imp = imp_num.fit_transform(data_num_Jersey_Number)
    return data_num_imp[mask_missing_values_only]
```

In [28]:
```
strategies[0], test_num_impute(strategies[0])
```

Out[28]:
```
('mean',
 array([19.54609577, 19.54609577, 19.54609577, 19.54609577, 19.54609577,
        19.54609577, 19.54609577, 19.54609577, 19.54609577, 19.54609577,
        19.54609577, 19.54609577, 19.54609577, 19.54609577, 19.54609577,
        19.54609577, 19.54609577, 19.54609577, 19.54609577, 19.54609577,
        19.54609577, 19.54609577, 19.54609577, 19.54609577, 19.54609577,
        19.54609577, 19.54609577, 19.54609577, 19.54609577, 19.54609577,
        19.54609577, 19.54609577, 19.54609577, 19.54609577, 19.54609577,
        19.54609577, 19.54609577, 19.54609577, 19.54609577, 19.54609577,
        19.54609577, 19.54609577, 19.54609577, 19.54609577, 19.54609577]))
```

In [29]:
```
strategies[1], test_num_impute(strategies[1])
```

Out[29]:
```
('median',
 array([17., 17., 17., 17., 17., 17., 17., 17., 17., 17., 17., 17.,
        17., 17., 17., 17., 17., 17., 17., 17., 17., 17., 17., 17.,
        17., 17., 17., 17., 17., 17., 17., 17., 17., 17., 17., 17.,
        17., 17., 17., 17., 17., 17., 17., 17., 17., 17., 17., 17.,
        17., 17., 17., 17., 17., 17., 17., 17.]))
```

In [30]:
```
strategies[2], test_num_impute(strategies[2])
```

Out[30]:
```
('most_frequent',
 array([8., 8., 8., 8., 8., 8., 8., 8., 8., 8., 8., 8., 8., 8., 8., 8.,
        8., 8., 8., 8., 8., 8., 8., 8., 8., 8., 8., 8., 8., 8., 8., 8.,
        8., 8., 8., 8., 8., 8., 8., 8., 8., 8., 8., 8., 8., 8., 8., 8.,
        8., 8., 8., 8., 8., 8., 8., 8.]))
```

Обработка пропусков в категориальных данных

In [34]:
```
cat_cols = []
for col in data.columns:
    # Количество пустых значений
    temp_null_count = data[data[col].isnull()].shape[0]
    dt = str(data[col].dtype)
    if temp_null_count>0 and (dt=='object'):
        cat_cols.append(col)
        temp_perc = round((temp_null_count / total_count) * 100.0, 2)
        print('Колонка {}. Тип данных {}. Количество пустых значений {}, {}%.'.format(col, dt, temp_null_count, temp_perc))
```

In [35]:
```
cat_temp_data = data[['Preferred Foot']]
cat_temp_data.head()
```

Out[35]:

	Preferred Foot
0	Left
1	Right
2	Right
3	Right
4	Right

In [36]:
```
cat_temp_data['Preferred Foot'].unique()
```

Out[36]:
```
array(['Left', 'Right', nan], dtype=object)
```

In [40]:
```
cat_temp_data[cat_temp_data['Preferred Foot'].isnull()].shape
```

Out[40]:
```
(48, 1)
```

In [41]:
```
imp2 = SimpleImputer(missing_values=np.nan, strategy='most_frequent')
data_imp2 = imp2.fit_transform(cat_temp_data)
data_imp2
```

Out[41]:
```
array([['Left'],
       ['Right'],
       ['Right'],
       ...,
       ['Right'],
       ['Right'],
       ['Right']], dtype=object)
```

In [42]:
```
np.unique(data_imp2)
```

Out[42]:
```
array(['Left', 'Right'], dtype=object)
```

In [43]:
```
imp3 = SimpleImputer(missing_values=np.nan, strategy='constant', fill_value='!!!')
data_imp3 = imp3.fit_transform(cat_temp_data)
data_imp3
```

Out[43]:
```
array([['Left'],
       ['Right'],
       ['Right'],
       ...,
       ['Right'],
       ['Right'],
       ['Right']], dtype=object)
```

In [44]:
```
np.unique(data_imp3)
```

Out[44]:
```
array(['!!!', 'Left', 'Right'], dtype=object)
```

In [45]:
```
data_imp3[data_imp3=='!!!'].size
```

Out[45]: 48

Преобразование категориальных признаков в числовые

In [46]:
```
cat_enc = pd.DataFrame({'c1':data_imp2.T[0]})
cat_enc
```

Out[46]:

	c1
0	Left
1	Right
2	Right
3	Right
4	Right
...	...
18202	Right
18203	Right
18204	Right
18205	Right
18206	Right

18207 rows × 1 columns

label encoding

In [47]:
```
from sklearn.preprocessing import LabelEncoder, OneHotEncoder
```

In [48]:
```
le = LabelEncoder()
cat_enc_le = le.fit_transform(cat_enc['c1'])
```

In [49]:
```
cat_enc['c1'].unique()
```

Out[49]: array(['Left', 'Right'], dtype=object)

In [50]:
```
np.unique(cat_enc_le)
```

Out[50]: array([0, 1])

In [52]:
```
le.inverse_transform([0, 1])
```

Out[52]: array(['Left', 'Right'], dtype=object)

one-hot encoding

In [53]:
```
ohe = OneHotEncoder()
cat_enc_ohe = ohe.fit_transform(cat_enc[['c1']])
```

In [54]:
```
cat_enc.shape
```

Out[54]:
```
(18207, 1)
```

In [55]:
```
cat_enc_ohe.shape
```

Out[55]:
```
(18207, 2)
```

In [56]:
```
cat_enc_ohe
```

Out[56]:
```
<18207x2 sparse matrix of type '<class 'numpy.float64'>'
        with 18207 stored elements in Compressed Sparse Row format>
```

In [57]:
```
cat_enc_ohe.todense()[0:10]
```

Out[57]:
```
matrix([[1., 0.],
        [0., 1.],
        [0., 1.],
        [0., 1.],
        [0., 1.],
        [0., 1.],
        [0., 1.],
        [0., 1.],
        [0., 1.],
        [0., 1.]])
```

In [58]:
```
cat_enc.head(10)
```

Out[58]:

	c1
0	Left
1	Right
2	Right
3	Right
4	Right
5	Right
6	Right
7	Right
8	Right
9	Right

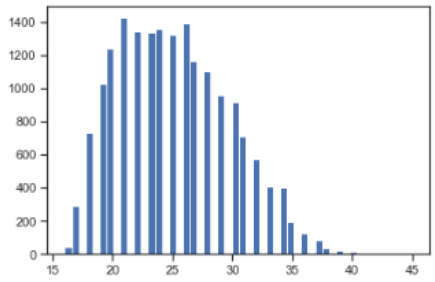
Масштабирование данных

In [59]:
```
from sklearn.preprocessing import MinMaxScaler, StandardScaler, Normalizer
```

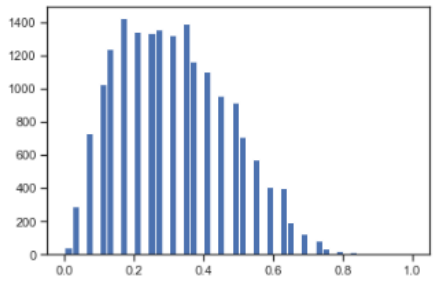
MinMax масштабирование

In [66]:
```
sc1 = MinMaxScaler()
sc1_data = sc1.fit_transform(data[['Age']])
```

In [67]:
```
plt.hist(data['Age'], 50)
plt.show()
```



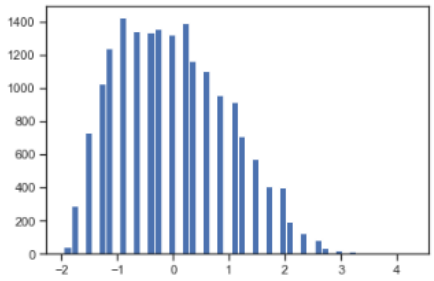
In [68]:
```
plt.hist(sc1_data, 50)
plt.show()
```



Масштабирование данных на основе Z-оценки

In [69]:
```
sc2 = StandardScaler()
sc2_data = sc2.fit_transform(data[['Age']])
```

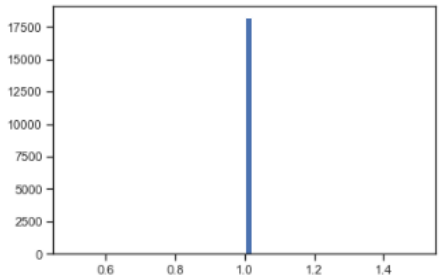
In [70]:
```
plt.hist(sc2_data, 50)
plt.show()
```



Нормализация данных

In [72]:
```
sc3 = Normalizer()
sc3_data = sc3.fit_transform(data[['Age']])
```

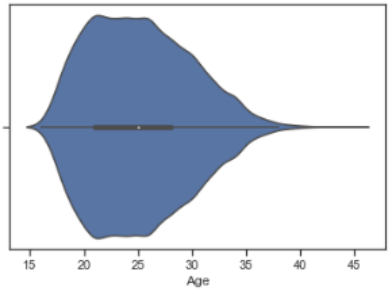
In [73]:
```
plt.hist(sc3_data, 50)
plt.show()
```



Violin plot

In [75]:
```
sns.violinplot(x=data['Age'])
```

Out[75]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x1e268785888>
```



Для количественных признаков использовался метод импутации. Для категориальных признаков использовались методы label encoding и one-hot encoding. Для дальнейшего построения ММО буду использовать метод Pandas get_dummies для категориальных признаков и импьютацию для числовых, так как pandas get dummies является быстрым вариантом one-hot кодирования, а импьютация меньше влияет на данные в целом и не изменит размер датасета по сравнению с удалением или заполнением нулями. Для масштабирования были использованы MinMax масштабирование и масштабирование данных на основе Z-оценки.