Московский государственный технический университет им. Н.Э. Баумана
Факультет «Информатика и системы управления»

Кафедра «Автоматизированные системы обработки информации и управления»

«Методы машинного обучения»

Отчет по Рубежному контролю №1

Выполнила:

студентка группы ИУ5-22М

Петрова Ирина

Проверил: доцент, к.т.н.

Гапанюк Ю. Е.

Москва, 2020

Рубежный контроль №1

Петрова Ирина ИУ5-22М

Вариант 11

Задача №2.

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?

Для студентов групп ИУ5-22М, ИУ5И-22М - для произвольной колонки данных построить гистограмму.

Набор данных: https://www.kaggle.com/karangadiya/fifa19 (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")
```

In [6]:

```
data = pd.read_csv('D:/Загрузки/fifa19/data.csv', sep=",")
data.head()
```

Out[6]:

	Unnamed: 0	ID	Name	Age	Photo	Nationality	
0	0	158023	L. Messi	31	https://cdn.sofifa.org/players/4/19/158023.png	Argentina	ht
1	1	20801	Cristiano Ronaldo	33	https://cdn.sofifa.org/players/4/19/20801.png	Portugal	ht
2	2	190871	Neymar Jr	26	https://cdn.sofifa.org/players/4/19/190871.png	Brazil	ht
3	3	193080	De Gea	27	https://cdn.sofifa.org/players/4/19/193080.png	Spain	ht
			K. De				

4	4	192985	27	https://cdn.sofifa.org/players/4/19/192985.png	Belgium	h Bruyne

5 rows × 89 columns

In [15]:

```
# размер набора данных
```

data.shape Out[15]:

```
(18207, 89)
```

In [16]:

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

```
Всего строк: 18207
```

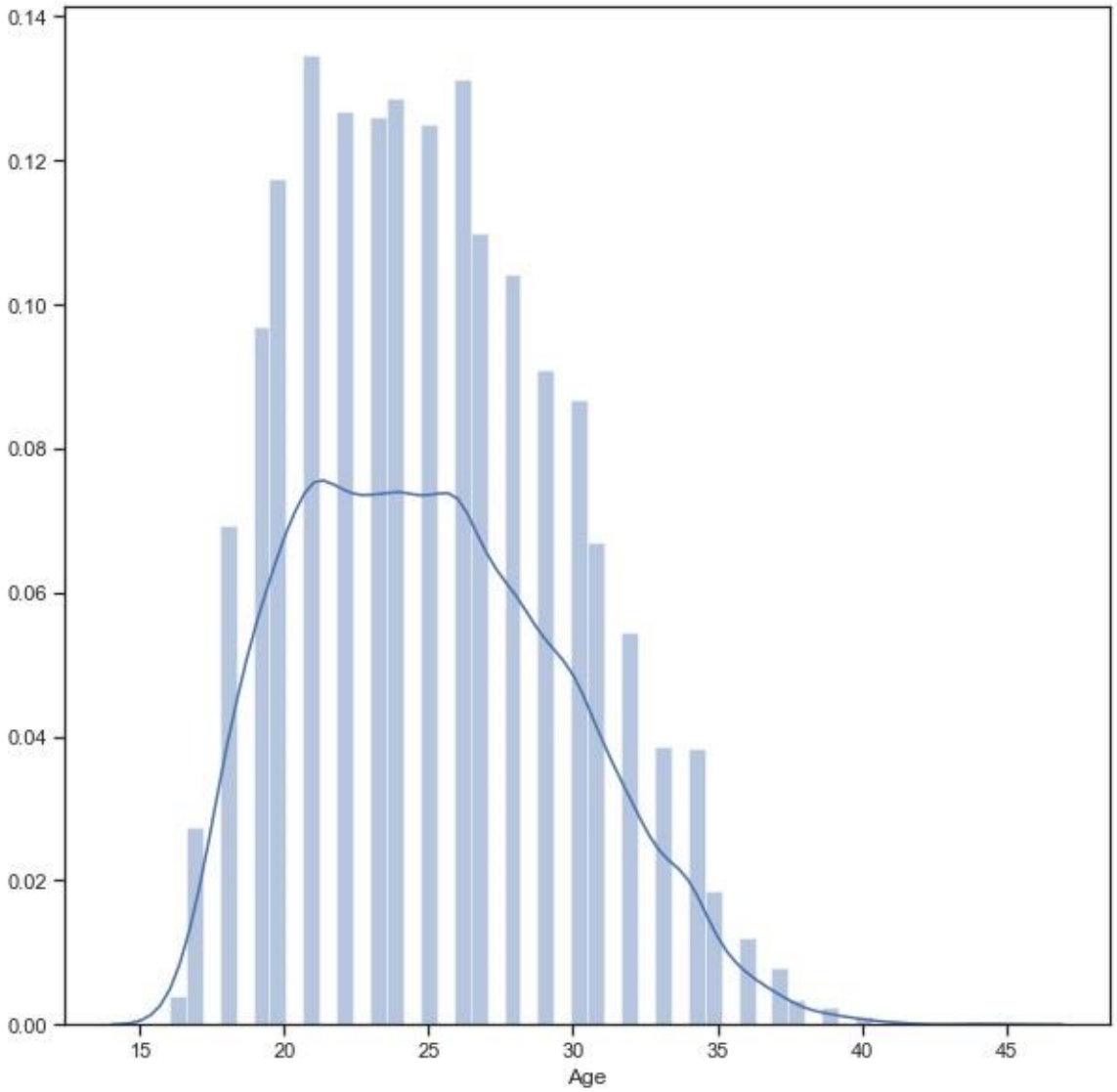
Построение гистограммы

In [4]:

```
# для колонки Age
fig, ax = plt.subplots(figsize=(10,10))
sns.distplot(data['Age'])
```

Out[4]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x21b1e9c6780>
```


0.14
0.12
0.10
0.08
0.06
0.04
0.02
0.00
15
20
25
30
35
40
45
Age

Обработка пропусков в данных

Простые стратегии - удаление или заполнение нулями

In [5]:

```
# проверим есть ли пропущенные значения
data.isnull().sum()
```

Out[5]:

```
Unnamed: 0           0
ID                   0
Name                 0
Age                  0
Photo                0
...
GKHandling          48
GKKicking           48
GKPositioning       48
GKReflexes          48
Release Clause    1564

Length: 89, dtype: int64
```

In [8]:

```
# Удаление колонок, содержащих пустые значения
data_new_1 = data.dropna(axis=1, how='any')
(data.shape, data_new_1.shape)
```

Out[8]: ((18207, 89),

(18207, 13)) In [9]:

```
# Удаление строк, содержащих пустые значения
data_new_2 = data.dropna(axis=0, how='any')
(data.shape, data_new_2.shape)
```

Out[9]:

```
((18207, 89), (0, 89))
```

In [13]:

```
# Заполнение всех пропущенных значений нулями
# В данном случае это некорректно, так как нулями заполняются в том числе категориальны
е колонки
data_new_3 = data.fillna(0)
data_new_3.head()
```

Out[13]:

	Unnamed: 0	ID	Name	Age	Photo	Nationality	
0	0	158023	L. Messi	31	https://cdn.sofifa.org/players/4/19/158023.png	Argentina	ht

1	1	20801	33	Cristiano Ronaldo	https://cdn.sofifa.org/players/4/19/20801.png	Portugal	ht	
2	2	190871	26	Neymar Jr	https://cdn.sofifa.org/players/4/19/190871.png	Brazil	ht	
3	3	193080	De Gea	27	https://cdn.sofifa.org/players/4/19/193080.png	Spain	ht	
4	4	192985	27	K. De Bruyne	https://cdn.sofifa.org/players/4/19/192985.png	Belgium	h	
5	rows × 89 columns							

"Внедрение значений" - импьютация (imputation)

Обработка пропусков в числовых данных

In [17]:

```
# Выберем числовые колонки с пропущенными значениями
# Цикл по колонкам датасета
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(c
ol, dt, temp_null_count, temp_perc))
```

Колонка International Reputation. Тип данных float64. Количество пустых зн
ачений 48, 0.26%.
Колонка Weak Foot. Тип данных float64. Количество пустых значений 48, 0.2
6%.
Колонка Skill Moves. Тип данных float64. Количество пустых значений 48, 0.
26%.
Колонка Jersey Number. Тип данных float64. Количество пустых значений 60,
0.33%.
Колонка Crossing. Тип данных float64. Количество пустых значений 48, 0.2
6%.
Колонка Finishing. Тип данных float64. Количество пустых значений 48, 0.2
6%.
Колонка HeadingAccuracy. Тип данных float64. Количество пустых значений 4
8, 0.26%.
Колонка ShortPassing. Тип данных float64. Количество пустых значений 48,
0.26%.
Колонка Volleys. Тип данных float64. Количество пустых значений 48, 0.26%.
Колонка Dribbling. Тип данных float64. Количество пустых значений 48, 0.2
6%.
Колонка Curve. Тип данных float64. Количество пустых значений 48, 0.26%.
Колонка FKAccuracy. Тип данных float64. Количество пустых значений 48, 0.2
6%.
Колонка LongPassing. Тип данных float64. Количество пустых значений 48, 0.
26%.
Колонка BallControl. Тип данных float64. Количество пустых значений 48, 0.
26%.
Колонка Acceleration. Тип данных float64. Количество пустых значений 48,
0.26%.
Колонка SprintSpeed. Тип данных float64. Количество пустых значений 48, 0.
26%.
Колонка Agility. Тип данных float64. Количество пустых значений 48, 0.26%.
Колонка Reactions. Тип данных float64. Количество пустых значений 48, 0.2
6%.
Колонка Balance. Тип данных float64. Количество пустых значений 48, 0.26%.
Колонка ShotPower. Тип данных float64. Количество пустых значений 48, 0.2
6%.
Колонка Jumping. Тип данных float64. Количество пустых значений 48, 0.26%.
Колонка Stamina. Тип данных float64. Количество пустых значений 48, 0.26%.
Колонка Strength. Тип данных float64. Количество пустых значений 48, 0.2
6%.
Колонка LongShots. Тип данных float64. Количество пустых значений 48, 0.2
6%.
Колонка Aggression. Тип данных float64. Количество пустых значений 48, 0.2
6%.
Колонка Interceptions. Тип данных float64. Количество пустых значений 48,
0.26%.
Колонка Positioning. Тип данных float64. Количество пустых значений 48, 0.
26%.
Колонка Vision. Тип данных float64. Количество пустых значений 48, 0.26%.
Колонка Penalties. Тип данных float64. Количество пустых значений 48, 0.2
6%.
Колонка Composure. Тип данных float64. Количество пустых значений 48, 0.2
6%.
Колонка Marking. Тип данных float64. Количество пустых значений 48, 0.26%.
Колонка StandingTackle. Тип данных float64. Количество пустых значений 48,
0.26%.

```
Колонка SlidingTackle. Тип данных float64. Количество пустых значений 48,
0.26%.
Колонка GKDiving. Тип данных float64. Количество пустых значений 48, 0.2
6%.
Колонка GKHandling. Тип данных float64. Количество пустых значений 48, 0.2
6%.
Колонка GKKicking. Тип данных float64. Количество пустых значений 48, 0.2
6%.
Колонка GKPositioning. Тип данных float64. Количество пустых значений 48,
0.26%.
Колонка GKReflexes. Тип данных float64. Количество пустых значений 48, 0.2
6%.
```

In [18]:

```
# Фильтр по колонкам с пропущенными значениями
data_num = data[num_cols]
data_num
```

Out[18]:

	International	Weak	Skill	Jersey			
	Foot	Moves	Number		Crossing	Finishing	HeadingAccuracy ShortPas Reputation
0	5.0	4.0	4.0	10.0	84.0	95.0	70.0
1	5.0	4.0	5.0	7.0	84.0	94.0	89.0
2	5.0	5.0	5.0	10.0	79.0	87.0	62.0
3	4.0	3.0	1.0	1.0	17.0	13.0	21.0
4	4.0	5.0	4.0	7.0	93.0	82.0	55.0
...	...	...	...	...	...	...	...
18202	1.0	2.0	2.0	22.0	34.0	38.0	40.0
18203	1.0	2.0	2.0	21.0	23.0	52.0	52.0
18204	1.0	3.0	2.0	33.0	25.0	40.0	46.0
18205	1.0	3.0	2.0	34.0	44.0	50.0	39.0
18206	1.0	3.0	2.0	33.0	41.0	34.0	46.0

18207 rows × 38 columns

In [24]:

```
# Фильтр по пустым значениям поля Agility
data[data['Agility'].isnull()]
```

Out[24]:

	Unnamed: 0	ID	Name	Age	Photo	Nation
13236	13236	177971	J. McNulty	33	https://cdn.sofifa.org/players/4/19/177971.png	Scot
13237	13237	195380	J. Barrera	29	https://cdn.sofifa.org/players/4/19/195380.png	Nicara
13238	13238	139317	J. Stead	35	https://cdn.sofifa.org/players/4/19/139317.png	Eng
13239	13239	240437	A. Semprini	20	https://cdn.sofifa.org/players/4/19/240437.png	
13240	13240	209462	R. Bingham	24	https://cdn.sofifa.org/players/4/19/209462.png	Eng
13241	13241	219702	K. Dankowski	21	https://cdn.sofifa.org/players/4/19/219702.png	Po
13242	13242	225590	I. Colman	23	https://cdn.sofifa.org/players/4/19/225590.png	Arge
13243	13243	233782	M. Feeney	19	https://cdn.sofifa.org/players/4/19/233782.png	Eng
13244	13244	239158	R. Minor	30	https://cdn.sofifa.org/players/4/19/239158.png	Denm
13245	13245	242998	Klauss	21	https://cdn.sofifa.org/players/4/19/242998.png	B
13246	13246	244022	I. Sissoko	22	https://cdn.sofifa.org/players/4/19/244022.png	Fr
13247	13247	189238	F. Hart	28	https://cdn.sofifa.org/players/4/19/189238.png	Au
13248	13248	211511	L. McCullough	24	https://cdn.sofifa.org/players/4/19/211511.png	Nort Ire
13249	13249	224055	Li Yunqiu	27	https://cdn.sofifa.org/players/4/19/224055.png	China
13250	13250	244535	F. Garcia	29	https://cdn.sofifa.org/players/4/19/244535.png	Parag
13251	13251	134968	R. Haemhouts	34	https://cdn.sofifa.org/players/4/19/134968.png	Belg
13252	13252	225336	E. Binaku	22	https://cdn.sofifa.org/players/4/19/225336.png	Alb
13253	13253	171320	G. Miller	31	https://cdn.sofifa.org/players/4/19/171320.png	Scot
13254	13254	246328	A. Aidonis	17	https://cdn.sofifa.org/players/4/19/246328.png	Germ
13255	13255	196921	L. Sowah	25	https://cdn.sofifa.org/players/4/19/196921.png	Germ
13256	13256	202809	R. Deacon	26	https://cdn.sofifa.org/players/4/19/202809.png	Eng

13257	13257	226617	Jang Hyun Soo	25	https://cdn.sofifa.org/players/4/19/226617.png		K Rep
S **13258**	13258	230713	A. Al Malki	23	https://cdn.sofifa.org/players/4/19/230713.png		Ar
13259	13259	234809	E. Guerrero	27	https://cdn.sofifa.org/players/4/19/234809.png		C

Unnamed: 0	**ID**	**Name**	**Age**	**Photo**	**Nation**
13260	13260	246073	Hernáiz	20	https://cdn.sofifa.org/players/4/19/246073.png	S
13261	13261	221498	H. Al Mansour	25	https://cdn.sofifa.org/players/4/19/221498.png	S Ar
13262	13262	244026	H. Paul	24	https://cdn.sofifa.org/players/4/19/244026.png	Germ
13263	13263	244538	S. Bauer	25	https://cdn.sofifa.org/players/4/19/244538.png	Au
13264	13264	201019	M. Chergui	29	https://cdn.sofifa.org/players/4/19/201019.png	Fr
13265	13265	221499	D. Gardner	28	https://cdn.sofifa.org/players/4/19/221499.png	Eng
13266	13266	237371	L. Bengtsson	20	https://cdn.sofifa.org/players/4/19/237371.png	Swe
13267	13267	242491	F. Jaramillo	22	https://cdn.sofifa.org/players/4/19/242491.png	Colo
13268	13268	153148	L. Garguła	37	https://cdn.sofifa.org/players/4/19/153148.png	Po
13269	13269	244540	S. Rivera	26	https://cdn.sofifa.org/players/4/19/244540.png	Colo
13270	13270	245564	Vinicius	19	https://cdn.sofifa.org/players/4/19/245564.png	B
13271	13271	213821	F. Sepúlveda	26	https://cdn.sofifa.org/players/4/19/213821.png	C
13272	13272	240701	L. Spence	22	https://cdn.sofifa.org/players/4/19/240701.png	Scot
13273	13273	242237	B. Lepistu	25	https://cdn.sofifa.org/players/4/19/242237.png	Est
13274	13274	244029	A. Abruscia	27	https://cdn.sofifa.org/players/4/19/244029.png	
13275	13275	244541	E. González	23	https://cdn.sofifa.org/players/4/19/244541.png	Venez
S **13276**	13276	211006	M. Al Amri	26	https://cdn.sofifa.org/players/4/19/211006.png	Ar
13277	13277	215102	J. Rebolledo	26	https://cdn.sofifa.org/players/4/19/215102.png	C

13278	13278	246078	C. Mamengi	17	https://cdn.sofifa.org/players/4/19/246078.png	Netherl	
13279	13279	239679	P. Mazzocchi	22	https://cdn.sofifa.org/players/4/19/239679.png		
13280	13280	244543	Y. Ammour	19	https://cdn.sofifa.org/players/4/19/244543.png	Fr	
13281	13281	212800	Jwa Joon Hyeop	27	https://cdn.sofifa.org/players/4/19/212800.png	K Rep	
13282	13282	231232	O. Marrufo	25	https://cdn.sofifa.org/players/4/19/231232.png	Me	
13283	13283	232256	Han Pengfei	25	https://cdn.sofifa.org/players/4/19/232256.png	China	

48 rows × 89 columns

In [25]:

```
# Запоминаем индексы строк с пустыми значениями
flt_index = data[data['Agility'].isnull()].index
flt_index
```

Out[25]:

```
Int64Index([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],
 dtype='int64')
```

In [27]:

```
# фильтр по колонке
data_num[data_num.index.isin(flt_index)]['Agility']
```

Out[27]:

```
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
13251   NaN
13252   NaN
13253   NaN
13254   NaN
13255   NaN
13256   NaN
13257   NaN
13258   NaN
13259   NaN
13260   NaN
13261   NaN
13262   NaN
13263   NaN
13264   NaN
13265   NaN
13266   NaN
13267   NaN
13268   NaN
13269   NaN
13270   NaN
13271   NaN
13272   NaN
13273   NaN
13274   NaN
13275   NaN
13276   NaN
13277   NaN
13278   NaN
13279   NaN
13280   NaN
13281   NaN
13282   NaN
13283   NaN
Name: Agility, dtype: float64 In [29]:
```

```
data_num_MasVnrArea = data_num[['Agility']]
data_num_MasVnrArea.head()
```

Out[29]:

	Agility
0	91.0
1	87.0
2	96.0
3	60.0
4	79.0

In [30]:

```
from sklearn.impute import SimpleImputer
from sklearn.impute import MissingIndicator
```

In [31]:

```
# Фильтр для проверки заполнения пустых значений
indicator = MissingIndicator()
mask_missing_values_only = indicator.fit_transform(data_num_MasVnrArea)
mask_missing_values_only Out[31]:
```

```
array([[False],
[False],
       [False],
...,
       [False],
       [False],
[False]])
```

In [36]:

```
def test_num_impute_col(dataset, column, strategy_param):
    temp_data = dataset[[column]]

    indicator = MissingIndicator()
    mask_missing_values_only = indicator.fit_transform(temp_data)

    imp_num = SimpleImputer(strategy=strategy_param)
 data_num_imp = imp_num.fit_transform(temp_data)

    filled_data = data_num_imp[mask_missing_values_only]

    return column, strategy_param, filled_data.size, filled_data[0], filled_data[filled
_data.size-1]
```

In [37]:

```
data[['Agility']].describe()
```

Out[37]:

	Agility
count	18159.000000
mean	63.503607
std	14.766049
min	14.000000
25%	55.000000
50%	66.000000
75%	74.000000
max	96.000000

In [38]:

```
test_num_impute_col(data, 'Agility', strategies[0])
```

Out[38]: ('Agility', 'mean', 48, 63.503607026818656,

63.503607026818656) In [39]:

```
test_num_impute_col(data, 'Agility', strategies[1])
```

Out[39]: ('Agility', 'median', 48,

66.0, 66.0) In [40]:

```
test_num_impute_col(data, 'Agility', strategies[2])
```

Out[40]:

('Agility', 'most_frequent', 48, 68.0, 68.0)

Обработка пропусков в категориальных данных

In [41]:

```
# Выберем категориальные колонки с пропущенными значениями
# Цикл по колонкам датасета
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(c
ol, dt, temp_null_count, temp_perc))
```

```
Колонка Club. Тип данных object. Количество пустых значений 241, 1.32%.
Колонка Preferred Foot. Тип данных object. Количество пустых значений 48,
0.26%.
Колонка Work Rate. Тип данных object. Количество пустых значений 48, 0.2
6%.
Колонка Body Type. Тип данных object. Количество пустых значений 48, 0.2
```

```
6%.
Колонка Real Face. Тип данных object. Количество пустых значений 48, 0.2
6%.
Колонка Position. Тип данных object. Количество пустых значений 60, 0.33%.
Колонка Joined. Тип данных object. Количество пустых значений 1553, 8.53%.
Колонка Loaned From. Тип данных object. Количество пустых значений 16943,
93.06%.
Колонка Contract Valid Until. Тип данных object. Количество пустых значени
й 289, 1.59%.
Колонка Height. Тип данных object. Количество пустых значений 48, 0.26%.
Колонка Weight. Тип данных object. Количество пустых значений 48, 0.26%.
Колонка LS. Тип данных object. Количество пустых значений 2085, 11.45%.
Колонка ST. Тип данных object. Количество пустых значений 2085, 11.45%.
Колонка RS. Тип данных object. Количество пустых значений 2085, 11.45%.
Колонка LW. Тип данных object. Количество пустых значений 2085, 11.45%.
Колонка LF. Тип данных object. Количество пустых значений 2085, 11.45%.
Колонка CF. Тип данных object. Количество пустых значений 2085, 11.45%.
Колонка RF. Тип данных object. Количество пустых значений 2085, 11.45%.
Колонка RW. Тип данных object. Количество пустых значений 2085, 11.45%.
Колонка LAM. Тип данных object. Количество пустых значений 2085, 11.45%.
Колонка CAM. Тип данных object. Количество пустых значений 2085, 11.45%.
Колонка RAM. Тип данных object. Количество пустых значений 2085, 11.45%.
Колонка LM. Тип данных object. Количество пустых значений 2085, 11.45%.
Колонка LCM. Тип данных object. Количество пустых значений 2085, 11.45%.
Колонка CM. Тип данных object. Количество пустых значений 2085, 11.45%.
Колонка RCM. Тип данных object. Количество пустых значений 2085, 11.45%.
Колонка RM. Тип данных object. Количество пустых значений 2085, 11.45%.
Колонка LWB. Тип данных object. Количество пустых значений 2085, 11.45%.
Колонка LDM. Тип данных object. Количество пустых значений 2085, 11.45%.
Колонка CDM. Тип данных object. Количество пустых значений 2085, 11.45%.
Колонка RDM. Тип данных object. Количество пустых значений 2085, 11.45%.
Колонка RWB. Тип данных object. Количество пустых значений 2085, 11.45%.
Колонка LB. Тип данных object. Количество пустых значений 2085, 11.45%.
Колонка LCB. Тип данных object. Количество пустых значений 2085, 11.45%.
Колонка CB. Тип данных object. Количество пустых значений 2085, 11.45%.
Колонка RCB. Тип данных object. Количество пустых значений 2085, 11.45%.
Колонка RB. Тип данных object. Количество пустых значений 2085, 11.45%.
Колонка Release Clause. Тип данных object. Количество пустых значений 156
4, 8.59%.
```

In [47]:

```
cat_temp_data = data[['Height']]
cat_temp_data.head()
```

Out[47]:

	Height
0	5'7
1	6'2
2	5'9
3	6'4
4	5'11

In [48]:

```
cat_temp_data['Height'].unique()
```

Out[48]:

```
array(["5'7", "6'2", "5'9", "6'4", "5'11", "5'8", "6'0", "5'6", "5'10",
       "6'6", "6'1", "5'4", "6'3", "5'5", "6'5", "6'7", "5'3", "5'2",
       "6'8", "5'1", "6'9", nan], dtype=object)
```

In [49]:

```
cat_temp_data[cat_temp_data['Height'].isnull()].shape
```

Out[49]:

```
(48, 1)
```

In [50]:

```
# Импьютация наиболее частыми значениями
imp2 = SimpleImputer(missing_values=np.nan, strategy='most_frequent')
data_imp2 = imp2.fit_transform(cat_temp_data) data_imp2 Out[50]:
```

```
array([["5'7"],
["6'2"],
       ["5'9"],
...,
       ["5'8"],
       ["5'10"],
       ["5'10"]], dtype=object)
```

In [51]:

```
# Пустые значения отсутствуют
np.unique(data_imp2)
```

Out[51]:

```
array(["5'1", "5'10", "5'11", "5'2", "5'3", "5'4", "5'5", "5'6", "5'7",
       "5'8", "5'9", "6'0", "6'1", "6'2", "6'3", "6'4", "6'5", "6'6",
       "6'7", "6'8", "6'9"], dtype=object)
```

In [53]:

```
# Импьютация константой
imp3 = SimpleImputer(missing_values=np.nan, strategy='constant', fill_value='!!!')
data_imp3 = imp3.fit_transform(cat_temp_data) data_imp3 Out[53]:
```

```
array([["5'7"],
["6'2"],
       ["5'9"],
...,
       ["5'8"],
       ["5'10"],
       ["5'10"]], dtype=object)
```

In [54]:

```
np.unique(data_imp3)
```

Out[54]:

```
array(['!!!', "5'1", "5'10", "5'11", "5'2", "5'3", "5'4", "5'5", "5'6",
       "5'7", "5'8", "5'9", "6'0", "6'1", "6'2", "6'3", "6'4", "6'5",
       "6'6", "6'7", "6'8", "6'9"], dtype=object)
```

In [55]:

```
data_imp3[data_imp3=='!!!'].size
```

Out[55]:

48

Какие способы обработки пропусков в данных для категориальных и количественных признаков Вы использовали? -- удаление строк и колонок с пустыми значениями, заполнение всех пропущенных значений нулями, импьютацию для количественных признаков и для категориальных (импьютация наиболее частыми значениями и константой)

Какие признаки Вы будете использовать для дальнейшего построения моделей машинного обучения и почему? -- для дальнейшего построения моделей будем использовать категориальные признаки со стратегиями "most_frequent" или "constant" для корректной работы класса SimpleImputer

Масштабирование данных

MinMax масштабирование

In [56]:

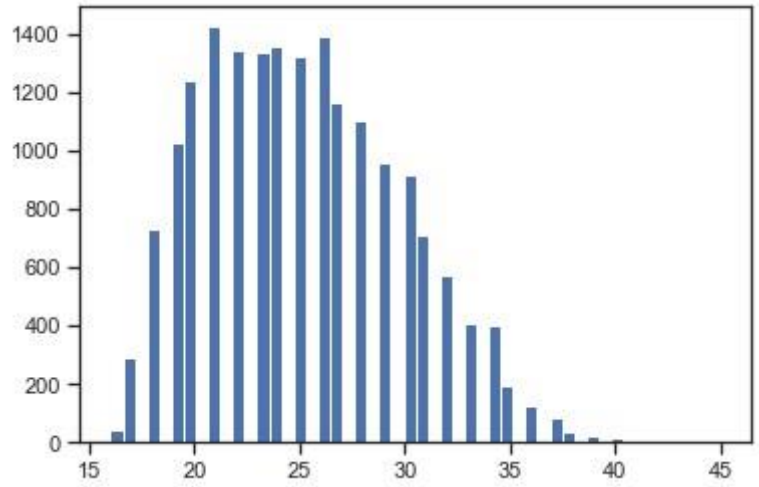
```
from sklearn.preprocessing import MinMaxScaler, StandardScaler,
```

Normalizer In [57]:

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

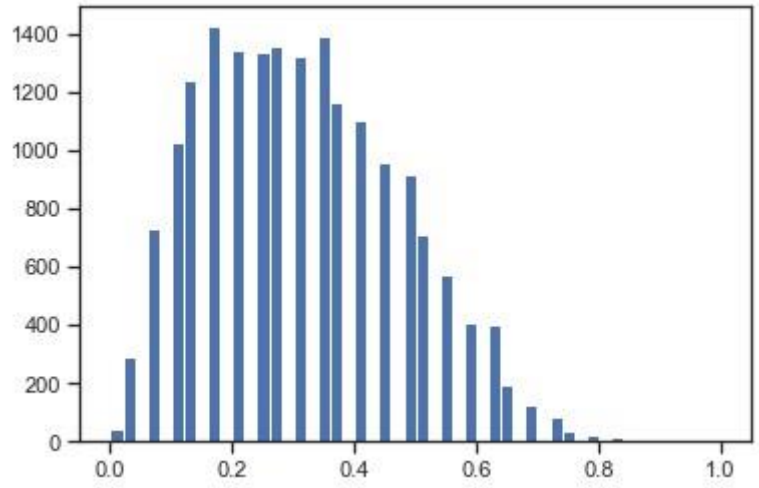
In [58]:

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



In [59]:

```
plt.hist(sc1_data, 50)
plt.show()
```



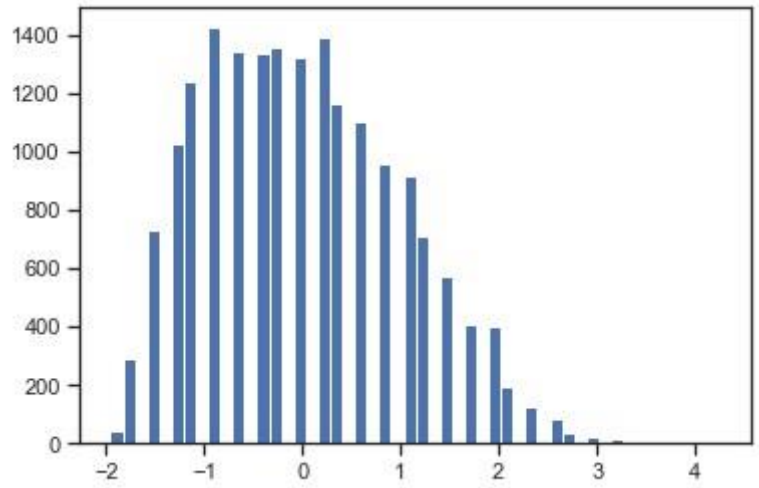
Масштабирование данных на основе Z-оценки - StandardScaler

In [60]:

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

In [61]:

```
plt.hist(sc2_data, 50)
plt.show()
```



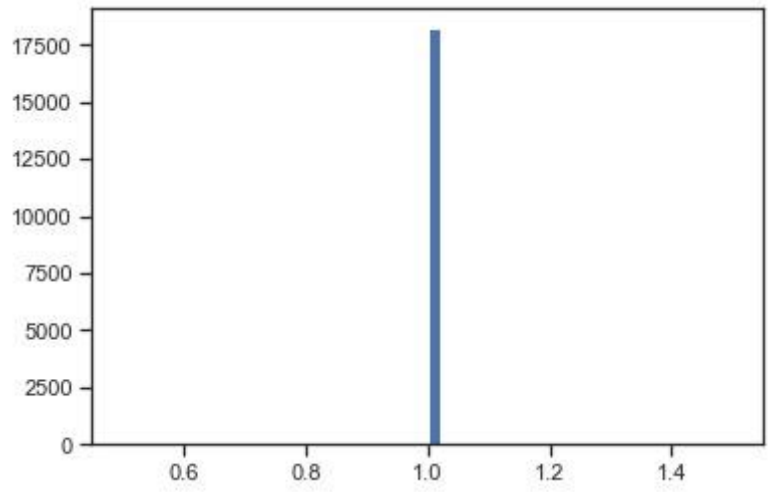
Нормализация данных

In [62]:

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

In [63]:

```
plt.hist(sc3_data, 50)
plt.show()
```



Преобразование категориальных признаков в количественные

In [69]:

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

Out[69]:

	c1
0	5'7
1	6'2
2	5'9
3	6'4
4	5'11
...	...
18202	5'9
18203	6'3
18204	5'8
18205	5'10
18206	5'10
18207	rows × 1 columns

Кодирование категорий целочисленными значениями - label encoding

In [70]:
```
from sklearn.preprocessing import LabelEncoder,
```

OneHotEncoder In [71]:

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

In [72]:

```
cat_enc['c1'].unique()
```

Out[72]:

```
array(["5'7", "6'2", "5'9", "6'4", "5'11", "5'8", "6'0", "5'6", "5'10",
       "6'6", "6'1", "5'4", "6'3", "5'5", "6'5", "6'7", "5'3", "5'2",
       "6'8", "5'1", "6'9"], dtype=object)
```

In [73]:

```
np.unique(cat_enc_le)
```

Out[73]:

```
array([ 0,  1,  2,  3,  4,  5,  6,  7,  8,  9, 10, 11, 12, 13, 14, 15, 16,
       17, 18, 19, 20])
```

In [75]:

```
le.inverse_transform([ 0,  1,  2,  3,  4,  5,  6,  7,  8,  9, 10, 11, 12, 13, 14, 15, 16,
       17, 18, 19, 20])
```

Out[75]:

```
array(["5'1", "5'10", "5'11", "5'2", "5'3", "5'4", "5'5", "5'6", "5'7",
       "5'8", "5'9", "6'0", "6'1", "6'2", "6'3", "6'4", "6'5", "6'6",
       "6'7", "6'8", "6'9"], dtype=object)
```

Кодирование категорий наборами бинарных значений - one-hot encoding

In [76]:

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

In [77]:

```
cat_enc.shape
```

Out[77]:

```
(18207, 1)
```

In [78]:

```
cat_enc_ohe.shape
```

Out[78]:

```
(18207, 21)
```

In [79]:

```
cat_enc_ohe
```

Out[79]:

```
<18207x21 sparse matrix of type '<class 'numpy.float64'>'    with
18207 stored elements in Compressed Sparse Row format>
```

In [80]:

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

Out[80]:

```
matrix([[0., 0., 0., 0., 0., 0., 0., 0., 1., 0., 0., 0., 0., 0., 0., 0.,
0., 0., 0., 0., 0.],
        [0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 1., 0., 0.,
0., 0., 0., 0., 0.],
        [0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 1., 0., 0., 0., 0., 0.,
0., 0., 0., 0., 0.],
        [0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 1.,
0., 0., 0., 0., 0.],
        [0., 0., 1., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
0., 0., 0., 0., 0.],
        [0., 0., 0., 0., 0., 0., 0., 0., 0., 1., 0., 0., 0., 0., 0., 0.,
0., 0., 0., 0., 0.],
        [0., 0., 0., 0., 0., 0., 0., 0., 0., 1., 0., 0., 0., 0., 0., 0.,
0., 0., 0., 0., 0.],
        [0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 1., 0., 0., 0., 0.,
0., 0., 0., 0., 0.],
        [0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 1., 0., 0., 0.,
0., 0., 0., 0., 0.],
        [0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 1., 0., 0.,
         0., 0., 0., 0., 0.]])
```

In [81]:

```
cat_enc.head(10)
```

Out[81]:

	c1
0	5'7
1	6'2
2	5'9
3	6'4

```
```

4	5'11
5	5'8
6	5'8
7	6'0
8	6'0
9	6'2

Pandas get_dummies - быстрый вариант one-hot кодирования

In [82]:

```
pd.get_dummies(cat_enc).head()
```

Out[82]:

	c1_5'1	c1_5'10	c1_5'11	c1_5'2	c1_5'3	c1_5'4	c1_5'5	c1_5'6	c1_5'7	c1_5'8	...	c1_6'0
0	0	0	0	0	0	0	0	0	1	0	...	0
1	0	0	0	0	0	0	0	0	0	0	...	0
2	0	0	0	0	0	0	0	0	0	0	...	0
3	0	0	0	0	0	0	0	0	0	0	...	0
4	0	0	1	0	0	0	0	0	0	0	...	0

5 rows × 21 columns

In [83]:

```
pd.get_dummies(cat_temp_data, dummy_na=True).head()
```

Out[83]:

	Height_5'1	Height_5'10	Height_5'11	Height_5'2	Height_5'3	Height_5'4	Height_5'5	Height
0	0	0	0	0	0	0	0	
1	0	0	0	0	0	0	0	
2	0	0	0	0	0	0	0	

3	0	0	0	0	0	0	0
4	0	0	1	0	0	0	0

5 rows × 22 columns