

Гаврилов Л.Я. ИУ5-63

Рубежный контроль №2

4 Вариант

Задание: Для заданного набора данных (по Вашему варианту) постройте модели классификации или регрессии (в зависимости от конкретной задачи, рассматриваемой в наборе данных). Для построения моделей используйте методы 1 и 2 (по варианту для Вашей группы). Оцените качество моделей на основе подходящих метрик качества (не менее двух метрик). Какие метрики качества Вы использовали и почему? Какие выводы Вы можете сделать о качестве построенных моделей? Для построения моделей необходимо выполнить требуемую предобработку данных: заполнение пропусков, кодирование категориальных признаков, и т.д.

Импорт библиотек:

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

```
In [2]: data = pd.read_csv("toy_dataset.csv")
data.shape
```

```
Out[2]: (150000, 6)
```

```
In [3]: data.head()
```

```
Out[3]:
```

	Number	City	Gender	Age	Income	Illness
0	1	Dallas	Male	41	40367.0	No
1	2	Dallas	Male	54	45084.0	No
2	3	Dallas	Male	42	52483.0	No
3	4	Dallas	Male	40	40941.0	No
4	5	Dallas	Male	46	50289.0	No

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

```
Out[4]: Number      int64
City      object
Gender     object
Age       int64
Income    float64
Illness    object
dtype: object
```

```
In [5]: # проверим есть ли пропущенные значения
data.isnull().sum()
```

```
Out[5]: Number      0
        City        0
        Gender      0
        Age         0
        Income      0
        Illness     0
        dtype: int64
```

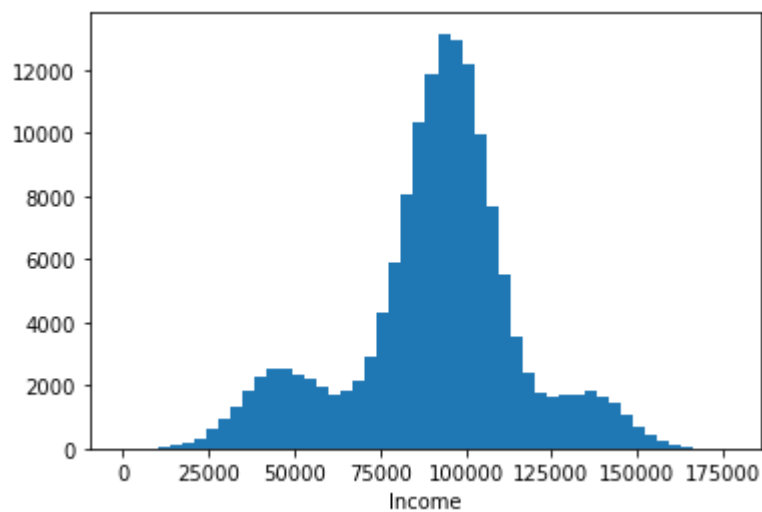
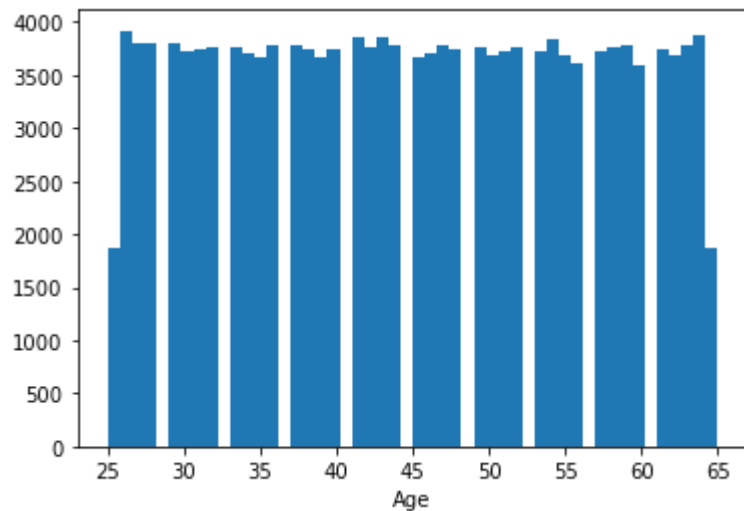
```
In [6]: # Определим уникальные значения для целевого признака
        data['Illness'].unique()
```

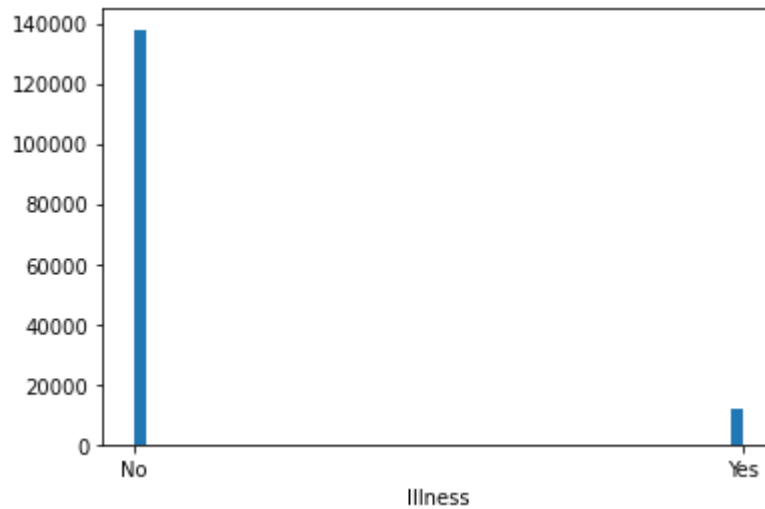
```
Out[6]: array(['No', 'Yes'], dtype=object)
```

Масштабирование данных:

```
In [7]: from sklearn.preprocessing import MinMaxScaler
```

```
In [8]: num_col = ['Age', 'Income', 'Illness']
        for col in data[num_col]:
            plt.hist(data[col], 50)
            plt.xlabel(col)
            plt.show()
```





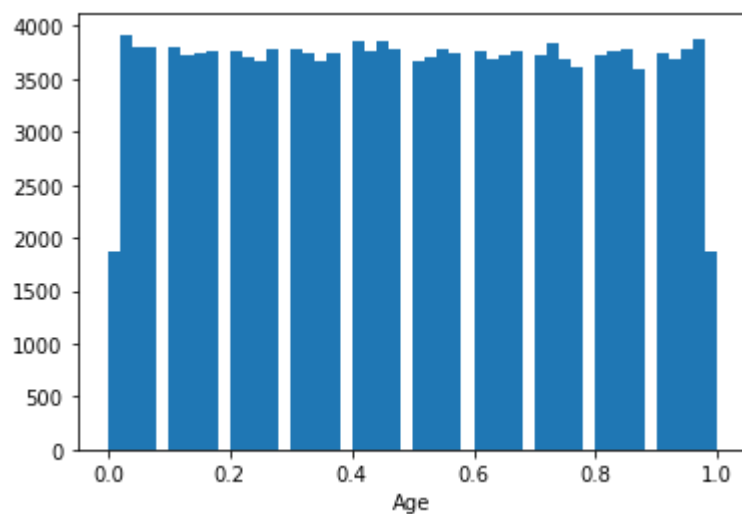
```
In [9]: from sklearn.preprocessing import LabelEncoder
```

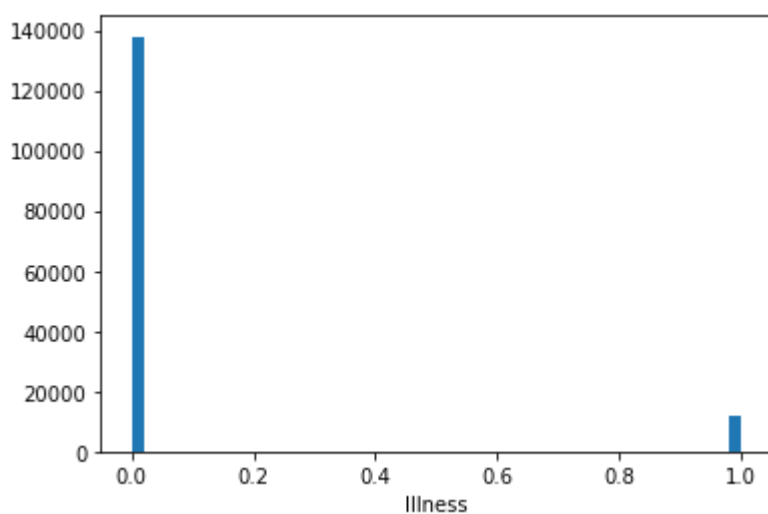
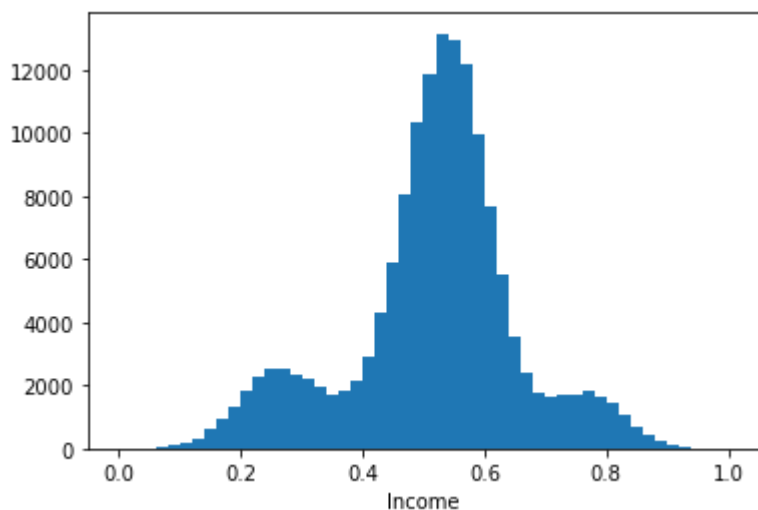
```
In [10]: le = LabelEncoder()
data.loc[:, 'Illness'] = le.fit_transform(data['Illness'])
data['Illness'].head()
```

```
Out[10]: 0    0
1    0
2    0
3    0
4    0
Name: Illness, dtype: int32
```

```
In [11]: # MinMax масштабирование
sc1 = MinMaxScaler()
for item in num_col:
    data.loc[:, item] = sc1.fit_transform(data[[item]])
```

```
In [12]: # Гистограмма по признакам, после масштабирования
for col in data[num_col]:
    plt.hist(data[col], 50)
    plt.xlabel(col)
    plt.show()
```





Кодирование категориальных признаков:

```
In [13]: cat_cols = ['City', 'Gender', 'Age']
one_hot = pd.get_dummies(data[cat_cols].astype(str))
one_hot.head()
```

```
Out[13]:
```

	City_Austin	City_Boston	City_Dallas	City_Los Angeles	City_Mountain View	City_New York City	City_San Diego	City
0	0	0	1	0	0	0	0	
1	0	0	1	0	0	0	0	
2	0	0	1	0	0	0	0	
3	0	0	1	0	0	0	0	
4	0	0	1	0	0	0	0	

5 rows × 51 columns

```
In [14]: # Замена бинарными значениями
data = data.join(one_hot)
data.drop(columns=cat_cols, inplace=True)
```

```
In [15]: data.head()
```

```
Out[15]:
```

Number	Income	Illness	City_Austin	City_Boston	City_Dallas	City_Los Angeles	City_Mountain View
--------	--------	---------	-------------	-------------	-------------	------------------	--------------------

0	1	0.230700	0.0	0	0	1	0
1	2	0.257228	0.0	0	0	1	0
2	3	0.298840	0.0	0	0	1	0
3	4	0.233928	0.0	0	0	1	0
4	5	0.286501	0.0	0	0	1	0

5 rows × 8 columns

Модели:

```
In [16]: from sklearn.model_selection import train_test_split
```

```
In [17]: data_train, data_test, data_y_train, data_y_test = train_test_split(data
```

Дерево решений:

```
In [18]: from sklearn.tree import DecisionTreeRegressor
```

```
In [19]: dtc = DecisionTreeRegressor(random_state=1).fit(data_train, data_y_train)
data_test_predicted_dtc = dtc.predict(data_test)
```

Случайный лес:

```
In [20]: from sklearn.ensemble import RandomForestRegressor
```

```
In [21]: RF = RandomForestRegressor(random_state=1).fit(data_train, data_y_train)
data_test_predicted_rf = RF.predict(data_test)
```

Оценка качества:

```
In [22]: from sklearn.metrics import mean_squared_error, r2_score
```

```
In [23]: # Mean squared error - средняя квадратичная ошибка
print('Метрика MSE:\nДерево решений: {}\nСлучайный лес: {}'.format(mean_
```

Метрика MSE:
Дерево решений: 196526557.47205406
Случайный лес: 118791341.57159367

```
In [24]: # 4) Метрика R2 или коэффициент детерминации
print('Метрика R\u00B2:\nДерево решений: {}\nСлучайный лес: {}'.format(r
```

Метрика R²:
Дерево решений: 0.8955637143042237
Случайный лес: 0.9368730280215708

Выводы о качестве построенных моделей:

Исходя из оценки качества построенных моделей можно увидеть, что модель "Случайный лес" лучше справляется с задачей по сравнению с моделью "Дерево решений", что может свидетельствовать о переобучении модели "Дерево решений".

```
In [ ]:
```

