



**Министерство науки и высшего образования Российской Федерации**  
**Федеральное государственное бюджетное образовательное учреждение**

**высшего образования**

**«Московский государственный технический университет**

**имени Н.Э. Баумана**

**(национальный исследовательский университет)»**

**(МГТУ им. Н.Э. Баумана)**

**Факультет «Информатика и системы управления»**

**Кафедра ИУ5 «Системы обработки информации и управления»**

**Курс «Технологии машинного обучения»**

**Отчет по лабораторной работе №4**

**Выполнила:**

**студент группы ИУ5-63Б**

**Латыпова К.Н.**

**Проверил:**

**преподаватель каф. ИУ5**

**Гапанюк Ю.Е.**

**Москва, 2021 г.**

## 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;
  - SVM;
  - дерево решений.
5. Оцените качество моделей с помощью двух подходящих для задачи метрик. Сравните качество полученных моделей.

## Текст программы и экранные формы:

### ЛР4

```
In [1]: import numpy as np
import pandas as pd
from typing import Dict, Tuple
import seaborn as sns
import matplotlib.pyplot as plt
%matplotlib inline
from sklearn.impute import SimpleImputer
import warnings
from sklearn.pipeline import Pipeline
from sklearn.preprocessing import PolynomialFeatures
from sklearn.metrics import confusion_matrix, precision_score, recall_score, f1_score, classification_report
from sklearn.linear_model import LinearRegression
warnings.simplefilter("ignore")
```

```
In [2]: # чтение обучающей выборки
data = pd.read_csv('letterdata.csv')
data.head()
```

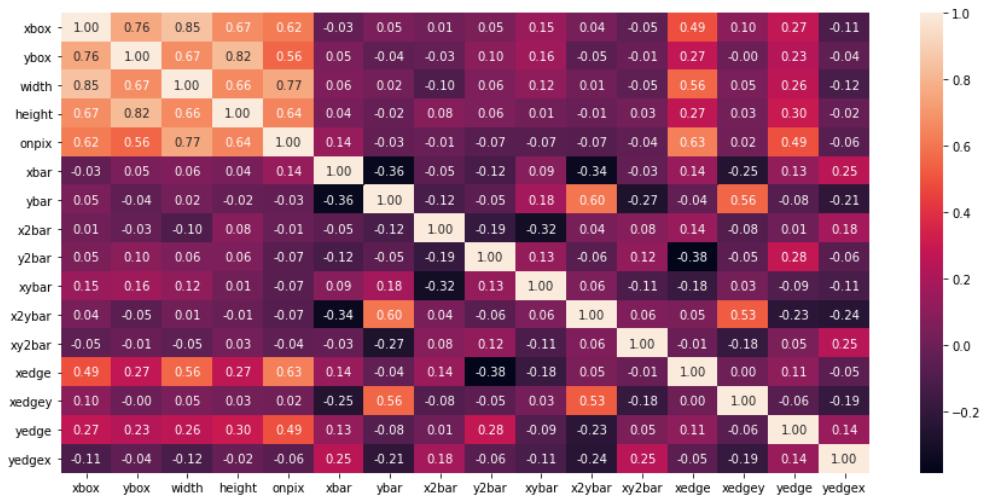
```
Out[2]:
```

|   | letter | xbox | ybox | width | height | onpix | xbar | ybar | x2bar | y2bar | xybar | x2ybar | xy2bar | xedge | xedgey | yedge | yedgex |
|---|--------|------|------|-------|--------|-------|------|------|-------|-------|-------|--------|--------|-------|--------|-------|--------|
| 0 | T      | 2    | 8    | 3     | 5      | 1     | 8    | 13   | 0     | 6     | 6     | 10     | 8      | 0     | 8      | 0     | 8      |
| 1 | I      | 5    | 12   | 3     | 7      | 2     | 10   | 5    | 5     | 4     | 13    | 3      | 9      | 2     | 8      | 4     | 10     |
| 2 | D      | 4    | 11   | 6     | 8      | 6     | 10   | 6    | 2     | 6     | 10    | 3      | 7      | 3     | 7      | 3     | 9      |
| 3 | N      | 7    | 11   | 6     | 6      | 3     | 5    | 9    | 4     | 6     | 4     | 4      | 10     | 6     | 10     | 2     | 8      |
| 4 | G      | 2    | 1    | 3     | 1      | 1     | 8    | 6    | 6     | 6     | 6     | 5      | 9      | 1     | 7      | 5     | 10     |

```
In [3]: from sklearn.model_selection import train_test_split
from sklearn.preprocessing import LabelEncoder
```

```
In [4]: #Построим корреляционную матрицу
fig, ax = plt.subplots(figsize=(15,7))
sns.heatmap(data.corr(method='pearson'), ax=ax, annot=True, fmt='.2f')
```

Out[4]: <AxesSubplot:>



```
B [5]: X = data.drop(['width', 'letter'], axis = 1)
Y = data.width
print('Входные данные:\n\n', X.head(), '\n\nВыходные данные:\n\n', Y.head())
```

Входные данные:

|   | xbox | ybox | height | onpix | xbar | ybar | x2bar | y2bar | xybar | x2ybar | xy2bar | \ |
|---|------|------|--------|-------|------|------|-------|-------|-------|--------|--------|---|
| 0 | 2    | 8    | 5      | 1     | 8    | 13   | 0     | 6     | 6     | 10     | 8      |   |
| 1 | 5    | 12   | 7      | 2     | 10   | 5    | 5     | 4     | 13    | 3      | 9      |   |
| 2 | 4    | 11   | 8      | 6     | 10   | 6    | 2     | 6     | 10    | 3      | 7      |   |
| 3 | 7    | 11   | 6      | 3     | 5    | 9    | 4     | 6     | 4     | 4      | 10     |   |
| 4 | 2    | 1    | 1      | 1     | 8    | 6    | 6     | 6     | 6     | 5      | 9      |   |

|   | xedge | xedgey | yedge | yedgex |
|---|-------|--------|-------|--------|
| 0 | 0     | 8      | 0     | 8      |
| 1 | 2     | 8      | 4     | 10     |
| 2 | 3     | 7      | 3     | 9      |
| 3 | 6     | 10     | 2     | 8      |
| 4 | 1     | 7      | 5     | 10     |

Выходные данные:

| 0 | 3 |
|---|---|
| 1 | 3 |
| 2 | 6 |
| 3 | 6 |
| 4 | 3 |

Name: width, dtype: int64

```
B [6]: X_train, X_test, Y_train, Y_test = train_test_split(X, Y, random_state = 0, test_size = 0.1)
print('Входные параметры обучающей выборки:\n\n', X_train.head(), \
      '\n\nВыходные параметры тестовой выборки:\n\n', X_test.head(), \
      '\n\nВыходные параметры обучающей выборки:\n\n', Y_train.head(), \
      '\n\nВыходные параметры тестовой выборки:\n\n', Y_test.head())
```

Входные параметры обучающей выборки:

|       | xbox | ybox | height | onpix | xbar | ybar | x2bar | y2bar | xybar | x2ybar | \ |
|-------|------|------|--------|-------|------|------|-------|-------|-------|--------|---|
| 17964 | 3    | 6    | 5      | 5     | 9    | 8    | 5     | 5     | 7     | 6      |   |
| 11632 | 2    | 1    | 1      | 1     | 6    | 9    | 8     | 4     | 7     | 5      |   |
| 10869 | 4    | 9    | 7      | 3     | 8    | 7    | 8     | 5     | 10    | 5      |   |
| 9179  | 4    | 10   | 8      | 2     | 7    | 9    | 0     | 8     | 14    | 6      |   |
| 8871  | 4    | 8    | 6      | 2     | 8    | 5    | 4     | 6     | 15    | 6      |   |

|       | xy2bar | xedge | xedgey | yedge | yedgex |
|-------|--------|-------|--------|-------|--------|
| 17964 | 8      | 5     | 8      | 9     | 11     |
| 11632 | 8      | 2     | 7      | 4     | 11     |
| 10869 | 9      | 3     | 8      | 4     | 7      |
| 9179  | 6      | 0     | 10     | 2     | 7      |
| 8871  | 11     | 1     | 6      | 0     | 7      |

Входные параметры тестовой выборки:

|       | xbox | ybox | height | onpix | xbar | ybar | x2bar | y2bar | xybar | x2ybar | \ |
|-------|------|------|--------|-------|------|------|-------|-------|-------|--------|---|
| 19134 | 3    | 3    | 2      | 1     | 4    | 11   | 2     | 7     | 11    | 10     |   |
| 4981  | 3    | 5    | 4      | 4     | 9    | 6    | 3     | 6     | 10    | 5      |   |
| 16643 | 4    | 8    | 5      | 2     | 3    | 8    | 8     | 2     | 7     | 5      |   |
| 19117 | 5    | 10   | 7      | 4     | 4    | 10   | 2     | 8     | 11    | 12     |   |
| 5306  | 4    | 7    | 8      | 5     | 8    | 7    | 6     | 4     | 8     | 7      |   |

|       | xy2bar | xedge | xedgey | yedge | yedgex |
|-------|--------|-------|--------|-------|--------|
| 19134 | 5      | 1     | 11     | 2     | 5      |
| 4981  | 7      | 2     | 8      | 5     | 9      |
| 16643 | 11     | 4     | 8      | 3     | 10     |
| 19117 | 9      | 3     | 9      | 2     | 6      |
| 5306  | 9      | 3     | 8      | 6     | 8      |

Выходные параметры обучающей выборки:

```
17964    5
11632    2
10869    4
9179     5
8871     5
Name: width, dtype: int64
```

Выходные параметры тестовой выборки:

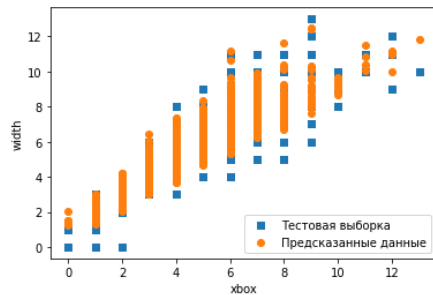
```
19134    4
4981     6
16643    4
19117    7
5306     4
Name: width, dtype: int64
```

```
B [7]: from sklearn.linear_model import LinearRegression
       from sklearn.metrics import mean_absolute_error, mean_squared_error, median_absolute_error, r2_score
```

```
B [8]: Lin_Reg = LinearRegression().fit(X_train, Y_train)

lr_y_pred = Lin_Reg.predict(X_test)
```

```
B [9]: plt.scatter(X_test.xbox, Y_test, marker = 's', label = 'Тестовая выборка')
       plt.scatter(X_test.xbox, lr_y_pred, marker = 'o', label = 'Предсказанные данные')
       plt.legend(loc = 'lower right')
       plt.xlabel ('xbox')
       plt.ylabel ('width')
       plt.show()
```



## SVM

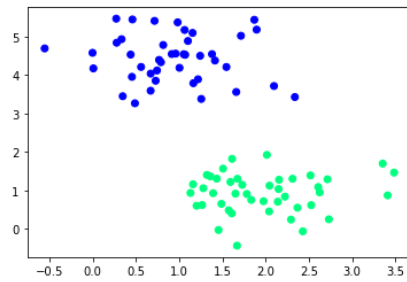
```
B [10]: from sklearn.svm import SVC , LinearSVC
        from sklearn.datasets.samples_generator import make_blobs
        from matplotlib import pyplot as plt
```

```
B [11]: X , y = make_blobs(n_samples=125, centers=2, cluster_std=0.6, random_state=0)
        # колич, кол кластеров, станд откл,

        train_X, test_X, train_y, test_y = train_test_split(X, y, test_size=40, random_state=0)

        plt.scatter(train_X[:, 0], train_X[:, 1], c=train_y, cmap='winter')
```

Out[11]: <matplotlib.collections.PathCollection at 0x21c668162e0>



```
B [12]: svc = SVC(kernel='linear')
        svc.fit(train_X, train_y)
```

Out[12]: SVC(kernel='linear')

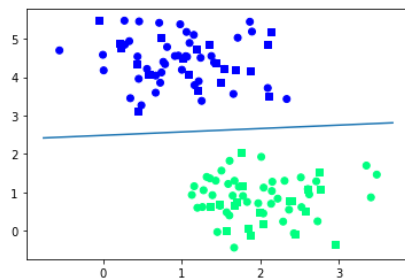
```
B [13]: plt.scatter(train_X[:, 0], train_X[:, 1], c=train_y, cmap='winter')

        ax=plt.gca()
        xlim=ax.get_xlim()

        ax.scatter(test_X[:, 0], test_X[:, 1], c=test_y, cmap='winter', marker='s')

        w= svc.coef_[0]
        a= -w[0]/w[1]
        xx=np.linspace(xlim[0], xlim[1])
        yy= a * xx - (svc.intercept_[0]/ w[1])
        plt.plot(xx, yy)
```

Out[13]: [<matplotlib.lines.Line2D at 0x21c665dc5b0>]



```
B [14]: pred_y = svc.predict(test_X)
```

```
B [15]: confusion_matrix(test_y, pred_y)
```

Out[15]: array([[21, 0],  
 [ 0, 19]], dtype=int64)

## Tree

```
B [16]: from sklearn.tree import DecisionTreeClassifier, DecisionTreeRegressor, export_graphviz
        from sklearn.tree import export_graphviz
        from sklearn import tree
        import re
```

```
B [17]: data = pd.read_csv('letterdata.csv')
        X = data.drop(['width', 'letter', 'onpix', 'xbar', 'ybar', 'x2bar', 'y2bar', 'xybar', 'x2ybar', 'xy2bar', 'xedge', 'xedge', 'yedge', 'y2edge', 'xyedge', 'xy2edge'], axis=1)
        Y = data.letter
        print('Входные данные:\n\n', X.head(), '\n\nВыходные данные:\n\n', Y.head())
```

Входные данные:

|   | xbox | ybox | height |
|---|------|------|--------|
| 0 | 2    | 8    | 5      |
| 1 | 5    | 12   | 7      |
| 2 | 4    | 11   | 8      |
| 3 | 7    | 11   | 6      |
| 4 | 2    | 1    | 1      |

Выходные данные:

|   |   |
|---|---|
| 0 | 3 |
| 1 | 3 |
| 2 | 6 |
| 3 | 6 |
| 4 | 3 |

Name: letter, dtype: int64

```
В [18]: # Обучим дерево на всех признаках iris
clf = tree.DecisionTreeClassifier()
clf = clf.fit(X, Y)
```

```
B [19]: from IPython.core.display import HTML
        from sklearn.tree.export import export_text
        tree_rules = export_text(clf, feature_names=list(X.columns))
        HTML('<pre>' + tree_rules + '</pre>')
```

```
Out[19]: |--- xbox <= 3.50
|         |--- xbox <= 2.50
|         |   |--- xbox <= 1.50
|         |   |   |--- height <= 3.50
|         |   |   |     |--- xbox <= 0.50
|         |   |   |     |   |--- ybox <= 0.50
|         |   |   |     |   |   |--- height <= 0.50
|         |   |   |     |   |   |   |--- class: 1
|         |   |   |     |   |   |   |--- height > 0.50
|         |   |   |     |   |   |   |   |--- class: 1
|         |   |   |     |   |   |   |--- ybox > 0.50
|         |   |   |     |   |   |   |   |--- height <= 1.50
|         |   |   |     |   |   |   |   |   |--- ybox <= 2.00
|         |   |   |     |   |   |   |   |   |   |--- class: 0
|         |   |   |     |   |   |   |   |   |   |--- ybox > 2.00
|         |   |   |     |   |   |   |   |   |   |   |--- class: 0
|         |   |   |     |   |   |   |   |   |   |   |   |--- height <= 1.50
```

```
B [20]: tree.plot_tree(clf)
```

```
Out[20]: [Text(103.96580247961957, 211.4, 'X[0] <= 3.5\nngini = 0.855\nnsamples = 20000\nnvalue = [195, 385, 1285, 1994, 3816, 4262, 364
1, 1946, 1418\n679, 237, 91, 39, 6, 4, 2]'),
Text(49.43657608695652, 199.32, 'X[0] <= 2.5\nngini = 0.766\nnsamples = 8459\nnvalue = [195, 385, 1284, 1955, 2961, 1503, 176,
0, 0, 0, 0\n0, 0, 0, 0, 0]'),
Text(26.45445652173913, 187.24, 'X[0] <= 1.5\nngini = 0.728\nnsamples = 4302\nnvalue = [195, 385, 1283, 1577, 841, 21, 0, 0, 0, 0,
0, 0, 0\n0, 0, 0, 0]'),
Text(15.082173913043478, 175.16, 'X[2] <= 3.5\nngini = 0.639\nnsamples = 1393\nnvalue = [189, 376, 714, 114, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0\n0, 0, 0, 0]'),
Text(6.388695652173913, 163.07999999999998, 'X[0] <= 0.5\nngini = 0.584\nnsamples = 1169\nnvalue = [54, 353, 655, 107, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0\n0, 0, 0, 0]'),
Text(2.911304347826087, 151.0, 'X[1] <= 0.5\nngini = 0.499\nnsamples = 99\nnvalue = [47, 52, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0\n0, 0]'),
Text(1.2939130434782609, 138.92000000000002, 'X[2] <= 0.5\nngini = 0.483\nnsamples = 76\nnvalue = [31, 45, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0\n0, 0, 0, 0]'),
Text(0.6469565217391304, 126.84, 'gini = 0.493\nnsamples = 59\nnvalue = [26, 33, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0\n0, 0]'),
Text(1.9408695652173913, 126.84, 'gini = 0.415\nnsamples = 17\nnvalue = [5, 12, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0\n0, 0]'),
Text(4.528695652173913, 138.92000000000002, 'X[2] <= 1.5\nngini = 0.423\nnsamples = 23\nnvalue = [16, 7, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0\n0, 0, 0, 0]'),
Text(3.234782608695652, 126.84, 'X[1] <= 2.0\nngini = 0.124\nnsamples = 15\nnvalue = [14, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0\n0, 0, 0, 0]')]
```