

УДК 004.9

## РЕАЛИЗАЦИЯ МОДЕЛИ МАККАЛЛОКА-ПИТТСА

Кустов А.С., старший оператор научной роты, I курс  
Военная академия связи имени Маршала Советского Союза С.М.  
Будённого, г. Санкт-Петербург

Реализуем программную модель МакКаллока-Питтса. Модель будет реализована без использования расширяющих программных библиотек (sklearn и других), поэтому необходимо создать отдельный класс программной модели [1] алгоритма машинного обучения (рисунок 1).

```
class MakKallok_Pitts(object):
    def __init__(self, eta=0.01, n_iter=50, random_state=1):
        self.eta = eta
        self.n_iter = n_iter
        self.random_state = random_state

    def fit(self, X, y):
        rgen = np.random.RandomState(self.random_state)
        self.w_ = rgen.normal(loc=0.0, scale=0.01, size=1 + X.shape[1])
        self.errors_ = []

        for _ in range(self.n_iter):
            errors = 0
            for xi, target in zip(X, y):
                update = self.eta * (target - self.predict(xi))
                self.w_[1:] += update * xi
                self.w_[0] += update
                errors += int(update != 0.0)
            self.errors_.append(errors)
        return self

    def net_input(self, X):
        #вычисление нового входа
        z = np.dot(X, self.w_[1:]) + self.w_[0]
        return z

    def predict(self, X):
        #возвращение метки класса после единичного шага
        return np.where(self.net_input(X) >= 0, 1, -1)
```

Рисунок 1 – Программная реализация модели МакКаллока-Питтса

Важным отличием модели МакКаллока-Питтса от других моделей машинного обучения является то, что её не нужно обучать, функция обобщения работает исключительно по пороговому значению, позволяя обновлять веса самостоятельно [2].

Оценкой точности разработанного решения будет анализ разделяющей поверхности цветков ириса по разным классам, которым они соответствуют [3].

Для этого нам, прежде всего, понадобится отделить классы цветков ириса из набора данных друг от друга (рисунок 2) [4].

```
# выбираем классы ириса
y = data.iloc[0:100, 4].values
y = np.where(y == 'Iris-setosa', -1, 1)

# получаем размеры лепестков и чашелистиков
X = data.iloc[0:100, [0, 2]].values

# отрисовываем данные в виде графического распределения образцов классов
plt.scatter(X[:50, 0], X[:50, 1],
            color='red', marker='o', label='setosa')
plt.scatter(X[50:100, 0], X[50:100, 1],
            color='blue', marker='x', label='versicolor')

plt.xlabel('sepal length [cm]')
plt.ylabel('petal length [cm]')
plt.legend(loc='upper left')

plt.show()
```

Рисунок 2 – Отделение классов цветков ириса по группам категориальных признаков

Результатом выполнения ячейки кода (рисунок 2) станет отрисованная геометрическая поверхность, внутри которой представлено соотношение отличающихся классов цветков ириса (рисунок 3) [5].

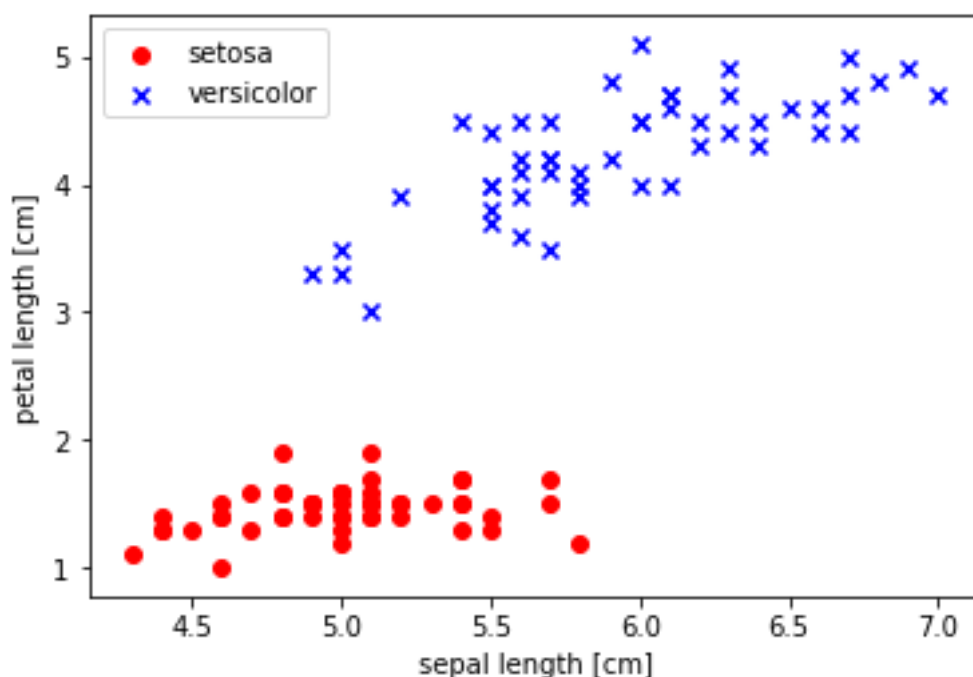


Рисунок 3 – Расположение объектов различных классов на геометрическом пространстве

Проверим, каким образом снижается ошибка классификации в модели МакКаллока-Питтса от итераций пересчета весов (эпох) [6] (рисунок 4). Результирующая диаграмма, представленная в виде графика, отображена на рисунке 5.

```
ppn = MakKallok_Pitts(eta=0.1, n_iter=10)

ppn.fit(X, y)

plt.plot(range(1, len(ppn.errors_) + 1), ppn.errors_, marker='o')
plt.show()
```

Рисунок 4 – Программное определение снижения ошибки от итераций пересчета весов

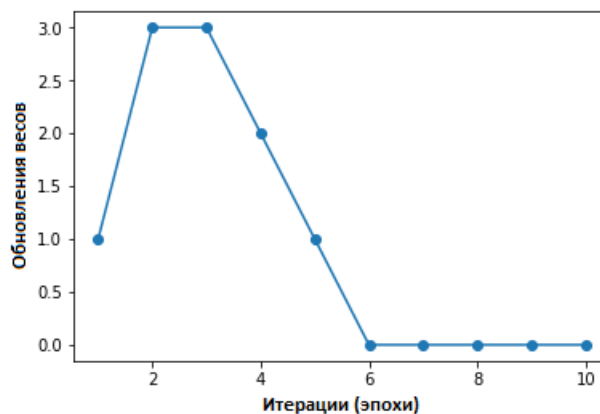


Рисунок 5 – Диаграмма зависимости снижения ошибки от итераций пересчета весов

Финальным шагом разработки модели машинного обучения является её оценка [7]. Прежде всего, в задаче классификации оценкой является качество классифицированных объектов одного класса по отношению к объектам другого класса – класс одного цветка не должен быть ошибочно присвоен другому классу.

На рисунках 6 – 7 представлена программная реализация проверки разработанного решения. Поскольку модель МакКаллока-Питтса реализовалась без сторонних библиотек, то и проверка также должна быть реализована «вручную», то есть самостоятельным написанием программного кода валидации решения [8].

```
def plot_decision_regions(X, y, classifier, resolution=0.02):

    markers = ('s', 'x', 'o', '^', 'v')
    colors = ('red', 'blue', 'lightgreen', 'gray', 'cyan')
    cmap = ListedColormap(colors[:len(np.unique(y))])
    x1_min, x1_max = X[:, 0].min() - 1, X[:, 0].max() + 1
    x2_min, x2_max = X[:, 1].min() - 1, X[:, 1].max() + 1
    xx1, xx2 = np.meshgrid(np.arange(x1_min, x1_max, resolution),
                           np.arange(x2_min, x2_max, resolution))
    Z = classifier.predict(np.array([xx1.ravel(), xx2.ravel()]).T)
    Z = Z.reshape(xx1.shape)
    plt.contourf(xx1, xx2, Z, alpha=0.3, cmap=cmap)
    plt.xlim(xx1.min(), xx1.max())
    plt.ylim(xx2.min(), xx2.max())
    for idx, cl in enumerate(np.unique(y)):
        plt.scatter(x=X[y == cl, 0],
                    y=X[y == cl, 1],
                    alpha=0.8,
                    c=colors[idx],
                    marker=markers[idx],
                    label=cl,
                    edgecolor='black')
```

Рисунок 6 – Реализация построения валидационной проверки модели МакКаллока-Питтса

```
plot_decision_regions(X, y, classifier=ppn)
plt.xlabel('sepal length [cm]')
plt.ylabel('petal length [cm]')
plt.legend(loc='upper left')
plt.show()
```

Рисунок 7 – Построение разделяющей гиперплоскости классов

На рисунке 7 представлена ячейка программного кода, выполнение которой позволяет получить геометрическое представление работы алгоритма (рисунок 8) – построение разделяющей гиперплоскости между классами [9].

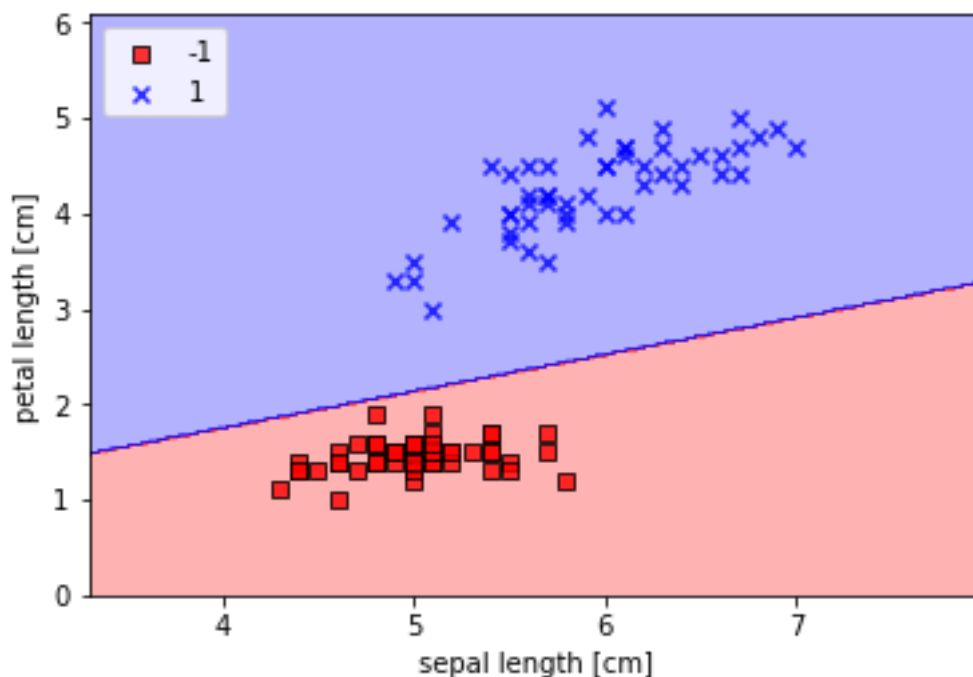


Рисунок 8 – Разделяющая гиперплоскость классов цветков ириса

Из рисунка 8 следует, что, на примере двух классов, разделяющая гиперплоскость была построена таким образом, чтобы все объекты класса «-1» были отнесены в одну стороны, а все объекты класса «1» присвоены другому классу [10]. Таким образом, точность разработанного решения на представленном наборе данных цветков ириса является достаточной для того, чтобы использовать разработанное решение в прикладных задачах [11].

#### Список литературы:

1. Свидетельство о государственной регистрации программы для ЭВМ № 2023680124 Российская Федерация. BrainPower : № 2023669010 : заявл. 16.09.2023 : опубл. 26.09.2023 / Р. В. Майтак. – EDN QXVJIM.
2. Математические и программные методы построения моделей глубокого обучения : Учебное пособие / А. В. Протодяконов и др. – Вологда : Общество с ограниченной ответственностью "Издательство "Инфра-Инженерия", 2023. – 176 с. – ISBN 978-5-9729-1484-5. – EDN PZLUAN.
3. Свидетельство о государственной регистрации программы для ЭВМ № 2023680335 Российская Федерация. Maitak Intelligence Natural Language Processing Module : № 2023669704 : заявл. 27.09.2023 : опубл. 28.09.2023 / Р. В. Майтак.
4. Методы восстановления непараметрической регрессии в условиях несбалансированных данных / А. Д. Салычева и др. – Вологда : Общество с ограниченной ответственностью "Издательство "Инфра-Инженерия", 2024. – 192 с. – ISBN 978-5-9729-1856-0. – EDN AAJATW.

5. Свидетельство о государственной регистрации программы для ЭВМ № 2023684619 Российская Федерация. Efficient Network: № 2023684038: заявл. 14.11.2023: опубл. 16.11.2023 / П. А. Пылов.
6. Свидетельство о государственной регистрации программы для ЭВМ № 2023680070 Российская Федерация. Модернизированная модель DBSCAN для определения скрытых взаимосвязей : № 2023668841 : заявл. 13.09.2023 : опубл. 26.09.2023 / Р. В. Майтак. – EDN KQUUKF.
7. Асимптотический анализ поведения прикладных моделей машинного обучения : Учебное пособие / А. В. Протодяконов и др. – Вологда : Общество с ограниченной ответственностью "Издательство "Инфра-Инженерия", 2023. – 144 с. – ISBN 978-5-9729-1455-5. – EDN APHQME.
8. Свидетельство о государственной регистрации программы для ЭВМ № 2023684621 Российская Федерация. Destructed Deep Random Forest: № 2023684050: заявл. 14.11.2023: опубл. 16.11.2023 / П. А. Пылов.
9. Свидетельство о государственной регистрации программы для ЭВМ № 2023684622 Российская Федерация. Mask Made AI: № 2023684042: заявл. 14.11.2023: опубл. 16.11.2023 / П. А. Пылов.
10. Свидетельство о государственной регистрации программы для ЭВМ № 2023680103 Российская Федерация. Cognitive Solution : № 2023669189 : заявл. 19.09.2023 : опубл. 26.09.2023 / Р. В. Майтак. – EDN QEMFJA.
11. Свидетельство о государственной регистрации программы для ЭВМ № 2023684624 Российская Федерация. Программа автоматического распознавания лиц в видеопотоке: № 2023684236: заявл. 15.11.2023: опубл. 16.11.2023 / П. А. Пылов.