Министерство науки и высшего образования
Федеральное государственное бюджетное образовательное
учреждение высшего образования
«Московский государственный технический университет
имени Н. Э. Баумана
(национальный исследовательский университет)»
(МГТУ им. Н. Э. Баумана)

Факультет «Фундаментальные науки»
Кафедра «Высшая математика»

ОТЧЕТ

ПО ОЗНАКОМИТЕЛЬНОЙ ПРАКТИКЕ

ЗА 5 СЕМЕСТР 2020—2021 ГОДА

Научный руководитель:
ст. преп. кафедры ФН1 _____ Кравченко О. В.
подпись, инициалы

студент группы ФН1–51Б _____ Сафонов А. А.
подпись, инициалы

Москва
2020

Содержание

1 Теоретическая часть

1.1 Бинарное дерево

Дерево – это структура, в которой у каждого узла может быть ноль или более подузлов – «детей». Например, дерево может выглядеть так:

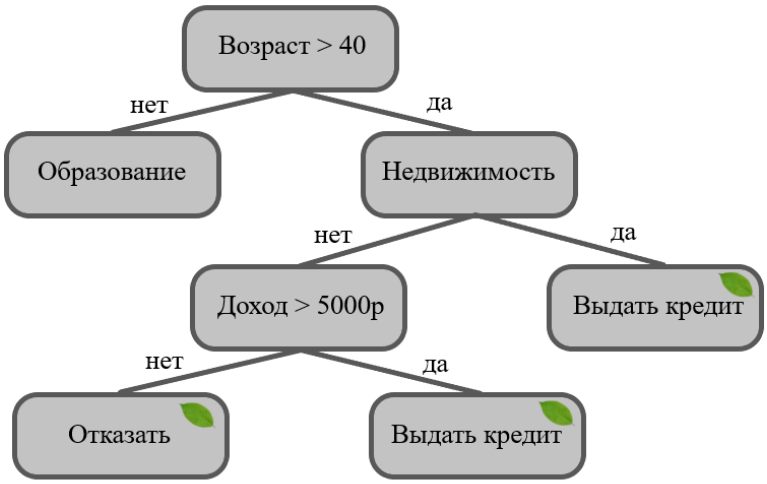

Рис. 1. Вариант бинарного дерева.

Двоичное дерево поиска похоже на дерево из примера выше, но строится по определенным правилам:

- У каждого узла не более двух детей(подузлов).
- Любое значение меньше значения узла становится левым ребенком(узлом).
- Любое значение больше или равное значению узла становится правым ребенком(узлом).

Решающие деревья используются для решения задач регрессии и классификации. Принцип заключается в следующем: алгоритм делит пространство признаков на подпространство, где для каждого объекта в этом подпространстве мы выдаем один результат. Для построения дерева необходимо определить его структуру. Каждой вершине соответствует свой признак и его пороговое значение, по которому мы будем сравниваться.

1.1.1 Критерий информативности

При построении дерева(подробнее с алгоритмом построения можно ознакомиться в статье [2]) необходимо задать функционал качества $Q(R_m, j, s)$, где j и s – порядковый номер элемента выборки и пороговое значение на текущем этапе соответственно, на основе которого осуществляется разбиение выборки на каждом шаге. Обозначим через R_m множество объектов, попавших в вершину, разбиваемую на данном шаге, а через R_ℓ и R_r – объекты, попадающие в левое и правое поддерево соответственно при заданном предикате. Мы будем использовать функционал, следующего вида:

$$Q(R_m, j, s) = H(R_m) - \frac{|R_\ell|}{R_m} H(R_\ell) - \frac{|R_r|}{R_m} H(R_r), \tag{1}$$

где $H(R)$ – критерий информативности, который оценивает качество распределения выборки на поддеревья. Наиболее распространенным критерием для решения задач классификации является энтропийный критерий вида

$$H(R) = \sum_{i=1}^{n} -p_i \log_2 p_i, \tag{2}$$

где p_i – частотная вероятность элемента/класса i наших данных.
Таким образом, чтобы максимизировать функционал качества $Q(R_m, j, s)$ выборки на каждом этапе разбиения, необходимо минимизировать сумму неоднородности 2–х выборок.

$$\frac{|R_\ell|}{R_m} H(R_\ell) + \frac{|R_r|}{R_m} H(R_r) \to \min \tag{3}$$

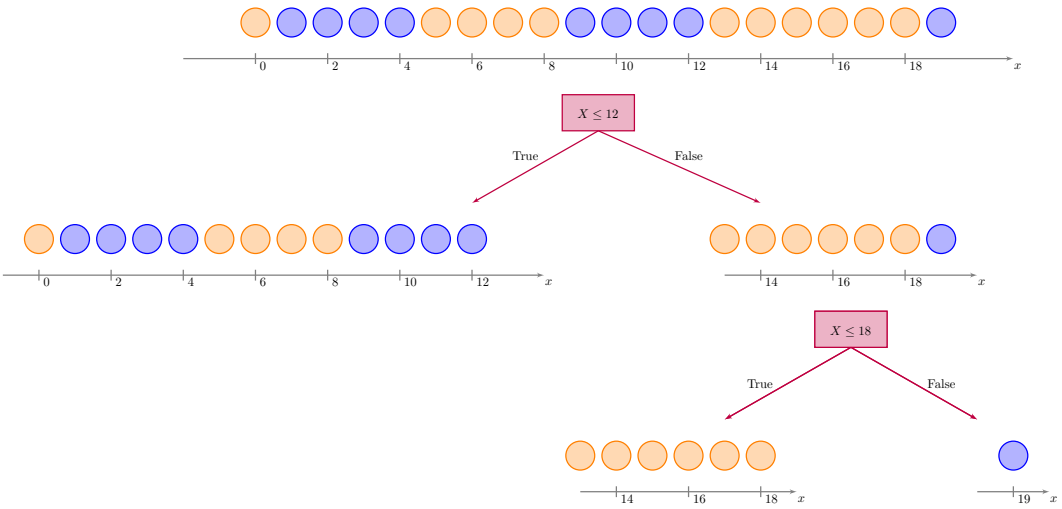

Рис. 2. Графическое представление разбиения исходной выборки по критерию информативности.

1.1.2 Построение решающего дерева

По итогам описания структуры решающего дерева можно сказать, что метод его построения определяется следующими параметрами:

1. определение предиката в каждом узле(признака, по которому на каждом этапе будет происходить разбиение выборки, а также его порогового значения)
2. функционал качества $Q(R_m, j, s)$ разбиения выборки
3. критерий останова(приведем несколько вариантов):
   - ограничение максимальной глубины дерева
   - ограничение минимального числа объектов в листе
   - ограничение максимального количества листьев в дереве
   - останов в случае, если все объекты в листе относятся к одному классу
   - Требование, что функционал качества при дроблении улучшался как минимум на t процентов

4. метод обработки пропущенных(NaN или None) значений

5. метод стрижки дерева(в случае, если некоторые ветви не являются достаточно информативными)

1.2 Метод опорных векторов

Данный метод обладает парой замечательных свойств:

1. обучение сводится к задаче квадратичного программирования, имеющей единственное решение

2. положение оптимальной разделяющей гиперплоскости зависит лишь от небольшой доли обучающих объектов, они и называются опорными векторами; остальные объекты фактически не задействуются.

1.2.1 Линейно разделимая выборка

Имеется обучающая выборка вида

$$X^\ell = (x_i, y_i)_{i=1}^\ell \tag{4}$$

Линейный пороговый классификатор:

$$a(x) = \text{sign}(\langle x, w\rangle - w_0), \tag{5}$$

где w и w_0 – вектор весов(направляющий вектор, разделяющий гиперплоскость) и какое–то скалярное значение(параллельный сдвиг гипервлоскости, разделяющий нашу выборку) соответственно.

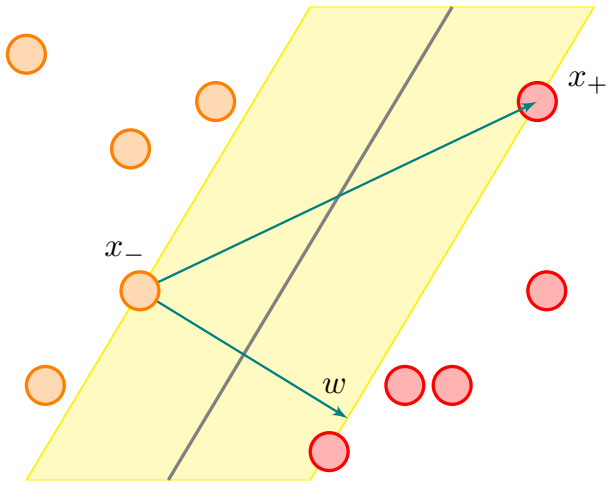

Рис. 3. Обучающие объекты x_+ и x_- находятся на границе разделяющей полосы. Вектор нормали w к разделяющей гиперплоскости определяет ширину полосы

Отступом(margin) объекта x_i от границы классов(гиперплоскости) является величина

$$M_i(w, w_0) = y_i(\langle x, w\rangle - w_0) \tag{6}$$

Алгоритм допускает ошибку на объекте x_i, если M_i отрицателен. Если $M_i \in (-1, +1)$, то объект попадает внутрь разделяющей полосы. Если же $M_i > 1$ – объект классифицируется правильно(находится на некотором расстоянии от разделяющей полосы).
Чтобы наша гиперплоскость была оптимальна, она должна максимально далеко отстоять от ближайших к ней точек разных классов. Для этого необходимо, чтобы выполнялось условие нормировки:

$$\min_{i=1,\dots,\ell} y_i(\langle x_i, w\rangle - w_0) = 1 \tag{7}$$

Таким образом, множество точек $\{x : -1 \leqslant \langle x, w\rangle - w_0 \leqslant 1\}$ – есть наша разделяющая полоса. Ширина такой полосы будет равна

$$\begin{cases} \langle x_+, w\rangle - w_0 = 1 \\ \langle x_-, w\rangle - w_0 = -1 \end{cases} \Rightarrow \frac{\langle x_+ - x_-, w\rangle}{||w||} = \frac{2}{||w||} \to \max \tag{8}$$

Отсюда следует, что w необходимо минимизировать по норме.
Таким образом, мы получаем задачу квадратичного программирования: требуется найти значения w и w_0, при которых выполняются ℓ ограничений–неравенств и норма w будет минимальной:

$$\begin{cases} \langle w, w\rangle \to \min \\ M_i(w, w_0) \geqslant 1, i = \overline{1, \ell} \end{cases} \tag{9}$$

1.2.2 Линейно неразделимая выборка

Чтобы обобщить постановку задачи на случай линейно неразделимой выборки, позволим алгоритму допускать ошибки на обучающих объектах, но при этом постараемся, чтобы ошибок было поменьше. В статье [1] представлена система ограничений вида

$$\begin{cases} \frac{1}{2}\langle w, w\rangle + C\sum_{i=1}^{\ell}\xi_i \to \min\limits_{w, w_0, \xi} \\ M_i(w, w_0) \geqslant 1 - \xi_i, i = \overline{1, \ell} \\ \xi_i \geqslant 0, i = \overline{1, \ell} \end{cases}, \tag{10}$$

где C – положительная константа, отвечает за максимизацию ширины разделяющей полосы и минимизацию суммарной ошибки. Ее обычно выбирают по *критерию скользящего контроля*. Это трудоёмкий способ, так как задачу приходится решать заново при каждом значении C.
Как мы могли уже заметить, в данном методе используется кусочно–линейная аппроксимация(график синего цвета) пороговой функии потерь(график красного цвета).
Таким образом, наша задача сводится к эквивалентной задаче минимизации функционала числа ошибок, имеющего вид

$$Q(w, w_0) = \sum_{i=1}^{\ell}(1 - M_i(w, w_0))_+ + \frac{1}{2C}||w^2|| \to \min_{w, w_0}, \tag{11}$$

где $(1 - M_i(w, w_0))_+$ – наша аппроксимирующая функция потерь, $||w^2||$ – регуляризатор L^2 и $\frac{1}{2C}$ – параметр регуляризации.

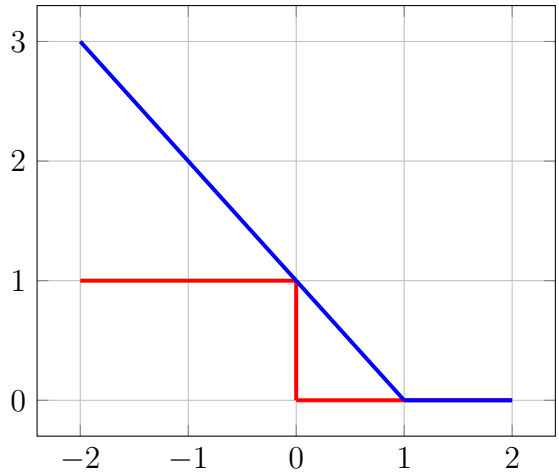

Рис. 4. Кусочно–линейная аппроксимация пороговой функции потерь

1.3 Логистическая регрессия

Метод логистической регрессии основан на довольно сильных вероятностных предположениях, которые имеют сразу несколько интересных последствий:

1. линейный алгоритм классификации оказывается оптимальным байесовским классификатором, основанным на применении теоремы Байеса:

$$P(H_i|A) = \frac{P(A|H_i)P(H_i)}{\sum_{i=1}^{n} P(A|H_i)P(H_i)} \tag{12}$$

2. определяется вид функции активации $\sigma(x) = \frac{1}{1+e^{-x}}$ и функции потерь $\Sigma(x) = \log_2(1 + e^{-x})$:

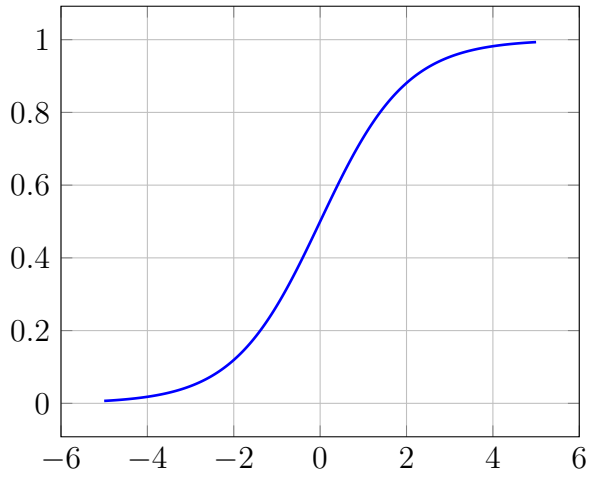

Рис. 5. Функция активации(Сигмоида)

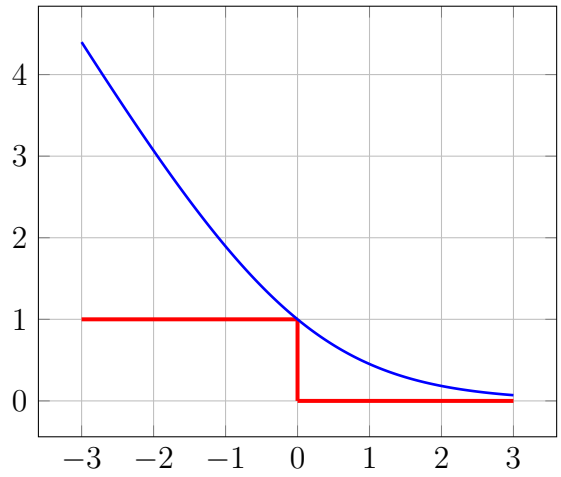

Рис. 6. Функция потерь

3. возможность наряду с классификацией объекта получать численную оценку вероятности его принадлежности каждому из классов

Имеется выборка $X^\ell = (x_i, y_i)_{i=1}^\ell$ и два класса $Y = \{-1, +1\}$. Объекты описываются n числовыми признаками $f_i : X \to \mathbb{R}, i = \overline{1, n}$. Выборка получена случайно согласно вероятностному распределению с плотностью

$$p(x, y) = P(y)p(x|y) = P(y|x)p(x), \tag{13}$$

где $P(y)$ – априорная вероятность, $p(x|y)$ – функция правдоподобия(плотность распределения выборки x в зависимости от конкретного класса y), $P(y|x)$ – апостериорная вероятность Y классов. Тогда оптимальный байесовский классификатор имеет вид

$$a(x) = \arg\max_{y \in Y} \lambda_y P(y|x) = \arg\max_{y \in Y} \lambda_y P(y)p(x|y), \tag{14}$$

где λ_y – штраф за ошибку на классах y.

1.3.1 Теорема

Если множество $X \times Y$ является вероятностным пространством, выборка $X^\ell = (x_i, y_i)_{i=1}^\ell$ получена случайно и независимо согласно вероятностному распределению с плотностью из уравнения (13), плотность распределения $p(x|y)$ относится к классу экспоненциальных (равномерное, нормальное, гипергеометрическое, пуассоновское, биномиальное, Г–распределение, и др.) распределений, а также среди признаков $f_1(x), ..., f_n(x)$ есть константа, то байесовский классификатор линеен:

$$a(x) = \text{sign}(\langle w, x \rangle - w_0), \qquad w_0 = \ln\left(\frac{\lambda_-}{\lambda_+}\right), \tag{15}$$

причем апостериорная вероятность отношения произвольного объекта x к конкретному классу y связана со значением дискриминантной функции:

$$P(y|x) = \sigma(\langle w, x \rangle y) \tag{16}$$

1.3.2 Принцип максимума правдоподобия

Используя теорему выше, с более подробным описанием и доказательством которой можно ознакомиться в статье [1], перейдем к функции правдоподобия. Для настройки вектора весов w необходимо максимизировать логарифм правдоподобия выборки:

$$L(w, X^\ell) = \log_2 \prod_{i=1}^{\ell} p(x_i, y_i) \to \max_w \tag{17}$$

Имеем $p(x, y) = P(y|x)p(x)$, где $p(x)$ – не зависит от параметра w. Апостериорная вероятность выражается по формуле (16) согласно теореме выше. Тогда

$$L(w) = \sum_{i=1}^{\ell} \log_2 \sigma(\langle w, x_i \rangle y_i) + \text{const}(w) \to \max_w \tag{18}$$

Таким образом, мы можем задачу максимизации $L(w)$ заменить на эквивалентную ей задачу минимизации функционала $Q(w)$:

$$Q(w) = \sum_{i=1}^{\ell} \log_2(1 + \exp(-\langle w, x_i \rangle y_i)) \to \min_w, \tag{19}$$

где $\langle w, x_i \rangle y_i = M_i(w)$ – отступ(margin) объекта x_i относительно объекта классификации $a(x, w) = \text{sign}\langle w, x \rangle$.

2 Практическая часть

2.1 Подготовка данных

Чтобы оценить работоспособность вышеописанных моделей классификации, необходимо в первую очередь подготовить наши тестируемые данные, то есть отобрать наиболее информативные признаки.

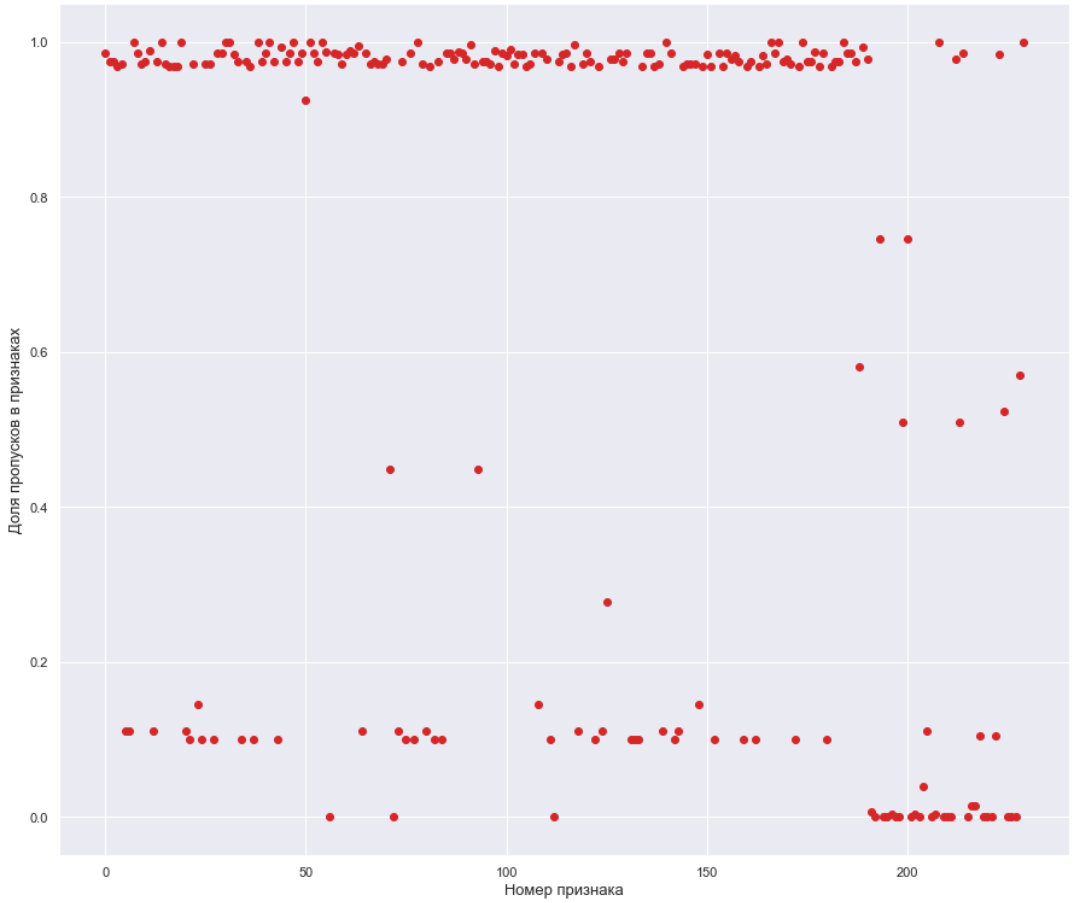

Рис. 7. Графическое представление информативности признаков

На рис. (7) можно заметить, что примерно 80% данных пустые. Таким образом, их можно не рассматривать, ибо они не содержат никакой полезной информации для обучения моделей. Тем не менее, большую часть оставшихся признаков можно считать вполне приемлимыми. Однако неоходимо провести небольшой анализ и по возможности дополнить их некой информацией.

Пропуски в категориальных данных заполним новыми категориальными значениями, а в вещественных – средним значением всех имеющихся данных соответствующего признака.

При отборе информативных признаков стоит обратить внимание на их тип:

1. для вещественных признаков в качестве корреляции воспользуемся разностью математических ожиданий
2. для категориальных признаков воспользуемся коэффициентом корреляции V–Крамера (данный критерий основан на статистике хи–квадрат с использованием таблицы сопряженностей признаков)

Таким образом, мы имеем два алгоритма для отбора признаков:

begin
  for *feature in dataset_features* **do**
    if *feature not empty* **then**
      correlation = $\mathbf{M}_{+}(feature) - \mathbf{M}_{-}(feature)$
    end
  end
end

Algorithm 1: Корреляция вещественных признаков

begin
  for *feature in dataset_features* **do**
    if *Nan values in feature > 1* **then**
      $\chi^2, p_{value} = \chi^2_{statistics}(congugation\_matrix(feature, target\_values))$
    end
  end
end

Algorithm 2: Корреляция категориальных признаков

Применим алгоритмы к данным, мы получим оптимальный набор достаточно информативных признаков для обучения и тестирования наших логических моделей.

Последнее, что следует определить – метрики, по которым мы будет оценивать качество классификации наших обработанных данных:

1. roc_auc – оценка качества классификации(определяет соотношение между долей объектов, верно классифицированных, к общему числу объектов)
2. recall – полнота классификации для каждого класса по отдельности
3. presicion – точность классификации для каждого класса по отдельности
4. f1 – среднее гармоническое между presicion и recall

2.2 Результаты

Обучение моделей проходило на 10000 различных объектах, а тестирование на 5000 объектах. Ниже представлена таблица с оценкой классификации по вышепредставленным метрикам обработанных данных при помощи **решающего дерева**, **метода опорных векторов** и **логистической регрессии**.

	Решающее дерево	Метод опорных векторов	Логистическая регрессия
roc_auc	0.600986	0.576267	0.599924
presicion	0.151840	0.000000	0.204545
recall	0.009228	0.000000	0.013175
f1	0.017348	0.000000	0.024654

Ссылка на практическую реализацию: {`https://github.com/AndroidSaf/Practice/blob/main/Practice.ipynb`}

Список литературы

[1] К. В. Воронцов, *Лекции по линейным алгоритмам классификации*, 2009, с. 1–43.

[2] Е. А. Соколов, *Решающие деревья*, 2018, с. 1–10.

[3] Орельен Жерон, Прикладное машинное обучение с помощью Scilit–Learn и TensorFlow: концепции, инструменты и техники для создания интеллектуальных систем. :Пер. с англ. — СпБ. :ООО «Альфа–книга», 2018.—688 с. : ил. — Парал. тит. англ.

[4] Aurelien Geron, Hands–on Machine Learning with Scikit–Learn, Keras, and TensorFlow, O'Reilly Media, Inc., 1005 Gravenstein Highway North, Sebastopol, CA 95472, 2019, pp. 510.

[5] Skilit–Learn [Электронный ресурс] URL: `https://scikit-learn.org/stable/index.html`. (Дата обращения: 14.01.2021)

[6] Pandas [Электронный ресурс] URL: `https://pandas.pydata.org/`. (Дата обращения: 15.01.2021)

[7] NumPy [Электронный ресурс] URL: `https://numpy.org/`. (Дата обращения: 13.01.2021)

[8] matplotlib [Электронный ресурс] URL: `https://matplotlib.org/`. (Дата обращения: 10.01.2021)

[9] К. В. Воронцов «Обзор постановок оптимизационных задач машинного обучения» [Электронный ресурс] URL: `https://www.youtube.com/watch?v=tX_MeIbfEmw`. (Дата обращения: 01.01.2021)

[10] Линейные методы классификации: метод опорных векторов – К. В. Воронцов [Электронный ресурс] URL: `https://www.youtube.com/watch?v=Adi67_94_gc`. (Дата обращения: 10.01.2021)

LaTeX