|  |  |
| --- | --- |
|  | **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 Системы обработки информации и управления .

**РАСЧЕТНО-ПОЯСНИТЕЛЬНАЯ ЗАПИСКА**

***К НАУЧНО-ИССЛЕДОВАТЕЛЬСКОЙ РАБОТЕ***

***НА ТЕМУ:***

***Модели машинного обучения***

***\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_***

***\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_***

***\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_***

***\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_***

Студент РТ5-61Б **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**  М. Ш. Нижаметдинов

(Группа) (Подпись, дата) (И.О.Фамилия)

Руководитель курсовой работы **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**  Ю. Е. Гапанюк

(Подпись, дата) (И.О.Фамилия)

Консультант **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

(Подпись, дата) (И.О.Фамилия)

*2023 г.*

**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)**

УТВЕРЖДАЮ

Заведующий кафедрой \_\_\_\_\_\_\_\_\_\_

(Индекс)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(И.О.Фамилия)

« \_\_\_\_\_ » \_\_\_\_\_\_\_\_\_\_\_\_ 20 \_\_\_\_ г.

**ЗАДАНИЕ**

**на выполнение научно-исследовательской работы**

по теме Модели машинного обучения .

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Студент группы РТ5-61Б

Нижаметдинов Мансур Шамилевич .

(Фамилия, имя, отчество)

Направленность НИР (учебная, исследовательская, практическая, производственная, др.)

учебная .

Источник тематики (кафедра, предприятие, НИР) НИР .

График выполнения НИР: 25% к 4 нед., 50% к 8 нед., 75% к 12 нед., 100% к 15 нед.

Техническое задание \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Решение задачи машинного обучения на основе материалов дисциплины . .

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

***Оформление научно-исследовательской работы:***

Расчетно-пояснительная записка на 32 листах формата А4.

Перечень графического (иллюстративного) материала (чертежи, плакаты, слайды и т.п.)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Дата выдачи задания « 7 » февраля 2023 г.

**Руководитель НИР**  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_ Ю. Е. Гапанюк

(Подпись, дата) (И.О.Фамилия)

**Студент \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_** \_ М. Ш. Нижаметдинов

(Подпись, дата) (И.О.Фамилия)

Примечание: Задание оформляется в двух экземплярах: один выдается студенту, второй хранится на кафедре.

Содержание

[Введение 4](#_Toc135408978)

[Основная часть 5](#_Toc135408979)

[Заключение 6](#_Toc135408980)

[Список использованных источников информации 7](#_Toc135408981)

[Приложение 8](#_Toc135408982)

# Введение

В современном мире машинное обучение является одной из наиболее перспективных и актуальных технологий, которая находит свое применение в различных сферах деятельности, начиная от медицины и финансов и заканчивая производством и транспортом. Технологии машинного обучения позволяют компьютерам обучаться на основе большого количества данных и использовать полученные знания для решения сложных задач. В данной научно-исследовательской работе рассмотрены основные принципы и методы машинного обучения. Мы изучим различные алгоритмы обучения, задачи классификации. В результате выполнения данной работы получены необходимые знания и навыки для работы с технологиями машинного обучения, что позволяет успешно применять эти технологии в практической деятельности.

# Основная часть

Цель научно-исследовательской работы – разработка эффективной модели машинного обучения для решения задачи классификации на выбранном наборе данных.

Последовательность действий:

1. Выбор набора данных для построения моделей машинного обучения.
2. Проведение разведочного анализа данных. Построение графиков, необходимых для понимания структуры данных. Анализ и заполнение пропусков в данных.
3. Выбор признаков, подходящих для построения моделей. Масштабирование данных.
4. Проведение корреляционного анализа данных. Формирование промежуточных выводов о возможности построения моделей.
5. Выбор метрик для последующей оценки качества моделей.
6. Выбор наиболее подходящих моделей для решения задачи классификации.
7. Формирование обучающей и тестовой выборок на основе исходных данных.
8. Построение базового решения для выбранных моделей без подбора гиперпараметров. Производится обучение моделей на основе обучающей выборки и оценка качества моделей на основе тестовой выборки.
9. Подбор гиперпараметров для выбранных моделей с помощью методов кросс-валидации.
10. Повторение пункта 8 для найденных оптимальных значений гиперпараметров. Сравнение качества полученных моделей.
11. Формирование выводов о качестве построенных моделей на основе выбранных метрик.
12. Создать веб-приложение для демонстрации хотя бы одной модели машинного обучения. У пользователя должна быть возможность изменения хотя бы одного гиперпараметра модели, при изменении гиперпараметра модель должна перестраиваться в веб-интерфейсе.

# Заключение

В результате проведенной научно-исследовательской работы была разработана эффективная модель машинного обучения для решения задачи классификации на выбранном наборе данных. В ходе работы были выполнены все поставленные задачи.

Полученные результаты позволяют сделать вывод о том, что построенные модели машинного обучения имеют высокое качество и могут быть использованы для решения задачи классификации на данном наборе данных. Веб-приложение для демонстрации модели машинного обучения позволяет пользователю изменять гиперпараметры модели и наблюдать за изменением ее качества в режиме реального времени.

Таким образом, научно-исследовательская работа по технологиям машинного обучения позволила успешно решить задачу классификации на выбранном наборе данных и создать веб-приложение для демонстрации модели машинного обучения. Полученные результаты могут быть использованы в различных областях, где требуется решение задач классификации на основе данных.

# Список использованных источников информации

1. Бурков, В.Н. Методы машинного обучения в задачах классификации / В.Н. Бурков. - М.: ФИЗМАТЛИТ, 2017. - 352 с.
2. Шестаков, А.В. Технологии машинного обучения: учебное пособие / А.В. Шестаков. - М.: Изд-во МГТУ им. Н.Э. Баумана, 2018. - 232 с.
3. Кузнецов, М.П. Машинное обучение и анализ данных: учебное пособие / М.П. Кузнецов, Е.В. Кузнецова. - М.: Изд-во МГУ, 2019. - 432 с.
4. Решетников, И.В. Методы машинного обучения и анализа данных: учебник для вузов / И.В. Решетников, В.К. Курганов, И.Б. Петров. - СПб.: БХВ-Петербург, 2018. - 480 с.
5. Карпов, О.В. Технологии машинного обучения: учебное пособие для студентов вузов / О.В. Карпов, М.В. Чернышев, А.В. Шестаков. - СПб.: Питер, 2019. - 288 с.

# Приложение

**Ход работы в Jupyter Notebook:**

## Поиск и выбор набора данных для построения моделей машинного обучения

В качестве набора данных используем набор данных химического анализа вин -

https://archive.ics.uci.edu/ml/machine-learning-databases/wine/wine.data

Данные являются результатом химического анализа вин, выращенных в одном и том же регионе Италии тремя разными культиваторами. Существует тринадцать различных измерений различных компонентов, содержащихся в трех типах вина.

Набор данных содержит следующие параметры:

Alcohol - Алкоголь;

Acid - Яблочная кислота;

Ash - Пепел;

Alcalinity of Ash - Щелочность пепла;

Magnesium - Магний;

Total Phenols - Всего фенолов;

Flavanoids - Флавоноиды;

Nonflavanoid Phenols - Нефлаваноидные фенолы;

Proanthocyanins - Проантоцианы;

Colour Intensity - Интенсивность цвета;

Hue - Оттенок;

OD280/OD315 of diluted wines - OD280/OD315 разбавленных вин;

Proline - Пролин.

### Импорт библиотек и загрузка датасета

import pandas as pd

import numpy as np

import seaborn as sns

import matplotlib.pyplot as plt

from sklearn.datasets import \*

from sklearn.preprocessing import MinMaxScaler

from sklearn.model\_selection import train\_test\_split

from sklearn.model\_selection import GridSearchCV

from sklearn.neighbors import KNeighborsClassifier

from sklearn.linear\_model import LogisticRegression

# from sklearn import svm, tree

from sklearn.tree import DecisionTreeClassifier

# from sklearn.ensemble import RandomForestClassifier

from sklearn.ensemble import BaggingClassifier

from sklearn.ensemble import GradientBoostingClassifier

from sklearn.svm import SVC

from sklearn.metrics import accuracy\_score, balanced\_accuracy\_score

from sklearn.metrics import roc\_curve, roc\_auc\_score

from sklearn.metrics import precision\_score, recall\_score, f1\_score, classification\_report

# from sklearn.metrics import mean\_absolute\_error

from sklearn.metrics import confusion\_matrix, ConfusionMatrixDisplay

from operator import itemgetter

def make\_dataframe(ds\_function):

ds = ds\_function()

df = pd.DataFrame(data= np.c\_[ds['data'], ds['target']],

columns= list(ds['feature\_names']) + ['target'])

return df

wine = load\_wine()

df = make\_dataframe(load\_wine)

## Проведение разведочного анализа данных. Построение графиков, необходимых для понимания структуры данных. Анализ и заполнение пропусков в данных.

### Основные характеристики датасета

# Первые 5 строк датасета

df.head()

# Размер датасета - 178 строк, 14 колонок

df.shape

# Список колонок

df.columns

# Список колонок с типами данных

df.dtypes

# Проверим наличие пустых значений

# Цикл по колонкам датасета

for col in df.columns:

# Количество пустых значений - все значения заполнены

temp\_null\_count = df[df[col].isnull()].shape[0]

print('{} - {}'.format(col, temp\_null\_count))

# Основные статистические характеристки набора данных

df.describe()

Выводы:

1. Все значения в датасете являются числовыми.

2. Представленный набор данных не содержит пропусков.

### Построение графиков для понимания структуры данных

sns.pairplot(df)

# Группировка по целевому признаку

sns.pairplot(df, hue="target")

# Убедимся, что целевой признак подходит для задачи классификации

df['target'].unique()

# Оценим дисбаланс классов для target

fig, ax = plt.subplots(figsize=(2,2))

plt.hist(df['target'])

plt.show()

df['target'].value\_counts()

# посчитаем дисбаланс классов

total = df.shape[0]

class\_0, class\_1, class\_2 = df['target'].value\_counts()

print('Класс 0 составляет {}%, класс 1 составляет {}%, а класс 2 составляет {}%'

.format(round(class\_0 / total, 4)\*100, round(class\_1 / total, 4)\*100, round(class\_2 / total, 4)\*100))

Вывод. Дисбаланс классов присутствует, но является приемлемым.

## Выбор признаков, подходящих для построения моделей. Кодирование категориальных признаков. Масштабирование данных. Формирование вспомогательных признаков, улучшающих качество моделей.

df.dtypes

Для построения моделей будем использовать все признаки.

Категориальные признаки отсутствуют, их кодирования не требуется.

Вспомогательные признаки для улучшения качества моделей в данном примере мы строить не будем.

Выполним масштабирование данных.

df.columns

data\_all=df

# колонки для масштабирования

scale\_cols = ['alcohol', 'malic\_acid', 'ash', 'alcalinity\_of\_ash', 'magnesium',

'total\_phenols', 'flavanoids', 'nonflavanoid\_phenols',

'proanthocyanins', 'color\_intensity', 'hue',

'od280/od315\_of\_diluted\_wines', 'proline']

sc1 = MinMaxScaler()

sc1\_data = sc1.fit\_transform(data\_all[scale\_cols])

# Добавим масштабированные данные в набор данных

for i in range(len(scale\_cols)):

col = scale\_cols[i]

new\_col\_name = col + '\_scaled'

data\_all[new\_col\_name] = sc1\_data[:,i]

data\_all.head()

# Проверим, что масштабирование не повлияло на распределение данных

for col in scale\_cols:

col\_scaled = col + '\_scaled'

fig, ax = plt.subplots(1, 2, figsize=(8,3))

ax[0].hist(data\_all[col], 50)

ax[1].hist(data\_all[col\_scaled], 50)

ax[0].title.set\_text(col)

ax[1].title.set\_text(col\_scaled)

plt.show()

## Проведение корреляционного анализа данных. Формирование промежуточных выводов о возможности построения моделей машинного обучения.

corr\_cols\_1 = scale\_cols + ['target']

corr\_cols\_1

df\_not\_scaled = data\_all[corr\_cols\_1]

df\_not\_scaled.head()

scale\_cols\_postfix = [x+'\_scaled' for x in scale\_cols]

corr\_cols\_2 = scale\_cols\_postfix + ['target']

corr\_cols\_2

df\_scaled = data\_all[corr\_cols\_2]

df\_scaled.head()

fig, ax = plt.subplots(1, 1, sharex='col', sharey='row', figsize=(15,5))

fig.suptitle('Корреляционная матрица (до масштабирования)')

sns.heatmap(df\_not\_scaled.corr(), ax=ax, annot=True, fmt='.3f')

fig, ax = plt.subplots(1, 1, sharex='col', sharey='row', figsize=(15,5))

fig.suptitle('Корреляционная матрица (после масштабирования)')

sns.heatmap(df\_scaled.corr(), ax=ax, annot=True, fmt='.3f')

Корреляционная матрица содержит коэффициенты корреляции между всеми парами признаков.

Корреляционная матрица симметрична относительно главной диагонали. На главной диагонали расположены единицы (корреляция признака самого с собой).

На основе корреляционной матрицы можно сделать следующие выводы:

1. Корреляционные матрицы для исходных и масштабированных данных совпадают.

2. Целевой признак наиболее сильно коррелирует с щелочностью пепла (0.52) и отрицательно коррелирует с флаваноидами (-0.85). Эти признаки обязательно следует оставить в модели.

3. Целевой признак слабо коррелирует с пеплом (-0.05). Скорее всего, этот признак стоит исключить из модели, возможно, он только ухудшит качество модели.

4. Целевой признак отчасти коррелирует с температурой (0.54). Этот признак стоит также оставить в модели.

5. Остальные признаки отчасти коррелируют как между собой, так и с целевым признаком. Их стои оставить в модели.

6. Большие по модулю значения коэффициентов корреляции свидетельствуют о значимой корреляции между исходными признаками и целевым признаком. На основании корреляционной матрицы можно сделать вывод о том, что данные позволяют построить модель машинного обучения.

## Выбор метрик для последующей оценки качества моделей

Для задачи классификации будем использовать следующие модели:

1. Логистическая регрессия

2. Метод ближайших соседей

3. Машина опорных векторов

4. Решающее дерево

5. Бэггинг

6. Градиентный бустинг

## Формирование обучающей и тестовой выборок на основе исходного набора данных

# На основе масштабированных данных выделим обучающую и тестовую выборки

y = df\_scaled['target']

x = df\_scaled.drop('target', axis = 1).drop('ash\_scaled', axis = 1)

x\_train, x\_test, y\_train, y\_test = train\_test\_split(x, y, test\_size = 0.4, random\_state = 7)

x\_train.shape, x\_test.shape

## Построение базового решения (baseline) для выбранных моделей без подбора гиперпараметров. Производится обучение моделей на основе обучающей выборки и оценка качества моделей на основе тестовой выборки.

# Модели

clas\_models = {'LogisticRegression': LogisticRegression(),

'KNN\_10':KNeighborsClassifier(n\_neighbors=10),

'SVC':SVC(probability=True),

'DecisionTree':DecisionTreeClassifier(),

'Bagging':BaggingClassifier(),

'GradientBoosting':GradientBoostingClassifier()}

class MetricLogger:

def \_\_init\_\_(self):

self.df = pd.DataFrame(

{'metric': pd.Series([], dtype='str'),

'alg': pd.Series([], dtype='str'),

'value': pd.Series([], dtype='float')})

def add(self, metric, alg, value):

"""

Добавление значения

"""

# Удаление значения если оно уже было ранее добавлено

self.df.drop(self.df[(self.df['metric']==metric)&(self.df['alg']==alg)].index, inplace = True)

# Добавление нового значения

temp = [{'metric':metric, 'alg':alg, 'value':value}]

self.df = self.df.append(temp, ignore\_index=True)

def get\_data\_for\_metric(self, metric, ascending=True):

"""

Формирование данных с фильтром по метрике

"""

temp\_data = self.df[self.df['metric']==metric]

temp\_data\_2 = temp\_data.sort\_values(by='value', ascending=ascending)

return temp\_data\_2['alg'].values, temp\_data\_2['value'].values

def plot(self, str\_header, metric, ascending=True, figsize=(5, 5)):

"""

Вывод графика

"""

array\_labels, array\_metric = self.get\_data\_for\_metric(metric, ascending)

fig, ax1 = plt.subplots(figsize=figsize)

pos = np.arange(len(array\_metric))

rects = ax1.barh(pos, array\_metric,

align='center',

height=0.5,

tick\_label=array\_labels)

ax1.set\_title(str\_header)

for a,b in zip(pos, array\_metric):

plt.text(0.5, a-0.05, str(round(b,3)), color='white')

plt.show()

# Сохранение метрик

clasMetricLogger = MetricLogger()

def clas\_train\_model(model\_name, model, clasMetricLogger):

model.fit(x\_train, y\_train)

# Предсказание значений

Y\_pred = model.predict(x\_test)

accuracy = accuracy\_score(y\_test.values, Y\_pred)

clasMetricLogger.add('accuracy', model\_name, accuracy)

fig, ax = plt.subplots(nrows=1, figsize=(10,5))

cm = confusion\_matrix(y\_test, Y\_pred, labels=np.unique(df\_scaled.target), normalize='true')

disp = ConfusionMatrixDisplay(confusion\_matrix=cm, display\_labels=np.unique(df\_scaled.target))

disp.plot(ax=ax)

ax.set\_title("Accuracy: {}".format(accuracy\_score(y\_test.values, Y\_pred)))

fig.suptitle(model\_name)

plt.show()

for model\_name, model in clas\_models.items():

clas\_train\_model(model\_name, model, clasMetricLogger)

## Подбор гиперпараметров для выбранных моделей. Рекомендуется использовать методы кросс-валидации. В зависимости от используемой библиотеки можно применять функцию GridSearchCV, использовать перебор параметров в цикле, или использовать другие методы.

x\_train.shape

n\_range = np.array(range(2,31,1))

tuned\_parameters = [{'n\_neighbors': n\_range}]

tuned\_parameters

%%time

clf\_gs = GridSearchCV(KNeighborsClassifier(), tuned\_parameters, cv=5, scoring='accuracy')

clf\_gs.fit(x\_train, y\_train)

# Лучшая модель

clf\_gs.best\_estimator\_

clf\_gs\_best\_params\_txt = str(clf\_gs.best\_params\_['n\_neighbors'])

clf\_gs\_best\_params\_txt

# Изменение качества на тестовой выборке в зависимости от К-соседей

plt.plot(n\_range, clf\_gs.cv\_results\_['mean\_test\_score'])

## Повторение для найденных оптимальных значений гиперпараметров. Сравнение качества полученных моделей с качеством baseline-моделей.

clas\_models\_grid = {'KNN\_10':KNeighborsClassifier(n\_neighbors=10),

str('KNN\_' + clf\_gs\_best\_params\_txt):clf\_gs.best\_estimator\_}

for model\_name, model in clas\_models\_grid.items():

clas\_train\_model(model\_name, model, clasMetricLogger)

## Формирование выводов о качестве построенных моделей на основе выбранных метрик. Результаты сравнения качества рекомендуется отобразить в виде графиков и сделать выводы в форме текстового описания.

# Метрики качества модели

clas\_metrics = clasMetricLogger.df['metric'].unique()

clas\_metrics

# Построим графики метрик качества модели

for metric in clas\_metrics:

clasMetricLogger.plot('Метрика: ' + metric, metric, figsize=(7, 6))

Вывод: лучшими оказались модели на основе метода опорных векторов и логистической регрессии.

prediction = model.predict(x\_test)

prediction

x\_test