|  |  |
| --- | --- |
|  | **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 «Автоматизированные системы обработки информации и управления»

**РАСЧЕТНО-ПОЯСНИТЕЛЬНАЯ ЗАПИСКА**

***К КУРСОВОЙ РАБОТЕ***

***НА ТЕМУ:***

***\_\_\_\_\_\_*** Решение комплексной задачи машинного обучения ***\_\_\_\_\_\_\_\_\_***

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

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

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

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

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



Студент \_\_\_\_\_ИУ5ц-82Б\_\_\_\_\_ **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_** \_\_\_Акимкин М.Г.\_\_\_\_

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

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

(Подпись, дата) (И.О.Фамилия)

Консультант **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_** \_\_\_Гапанюк Ю.Е.\_\_\_\_

(Подпись, дата) (И.О.Фамилия)

*2021 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)

« \_\_\_\_\_ » \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 2021 г.

**ЗАДАНИЕ**

**на выполнение курсовой работы**

по дисциплине \_\_\_\_ Технологии машинного обучения\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Студент группы \_\_\_\_\_\_ИУ5ц-82Б\_\_\_\_\_\_\_\_\_

\_\_\_\_Акимкин Максим Григорьевич\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(Фамилия, имя, отчество)

Тема курсовой работы \_\_\_\_ решение комплексной задачи машинного обучения \_\_\_\_\_\_\_\_\_\_

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

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

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

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

Источник тематики (кафедра, предприятие, НИР) \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

График выполнения работы: 25% к \_\_\_ нед., 50% к \_\_\_ нед., 75% к \_\_ нед., 100% к \_\_\_ нед.

***Задание*** \_\_\_\_\_\_ решение задачи машинного обучения на основе материалов дисциплины. Выполняется студентом единолично. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

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

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

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

***Оформление курсовой работы:***

Расчетно-пояснительная записка на 12 листах формата А4.

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

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

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

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

Дата выдачи задания « 10» мая 2021 г.

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

(Подпись, дата) (И.О.Фамилия)

**Студент \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_** \_\_\_Акимкин М.Г.\_\_\_\_\_



(Подпись, дата) (И.О.Фамилия)

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

Запуск курсового проекта:

1. Открыть в терминале директорию проекта.

2. Прописать в терминале команду: streamlit run 6lab.py.

Прим.: Название ”6lab” осталось исторически, исходя из хода разработки проекта.

3. Откроется браузер с запущенным проектом.

Прим.: Если браузер не открылся автоматически, необходимо перейти по адресу, который был выведен в терминал; адреса в терминале два:

Пример вывода в терминале:

”You can now view your Streamlit app in your browser.

Local URL: http://localhost:8501

Network URL: http://172.16.93.210:8501”

Можно перейти, как по первому (локальный – более предпочтителен), так и по второму (сетевой).

Работа в курсовом проекте:

1. Браузер откроется на ”основном окне проекта” (*см. Рис.1*):

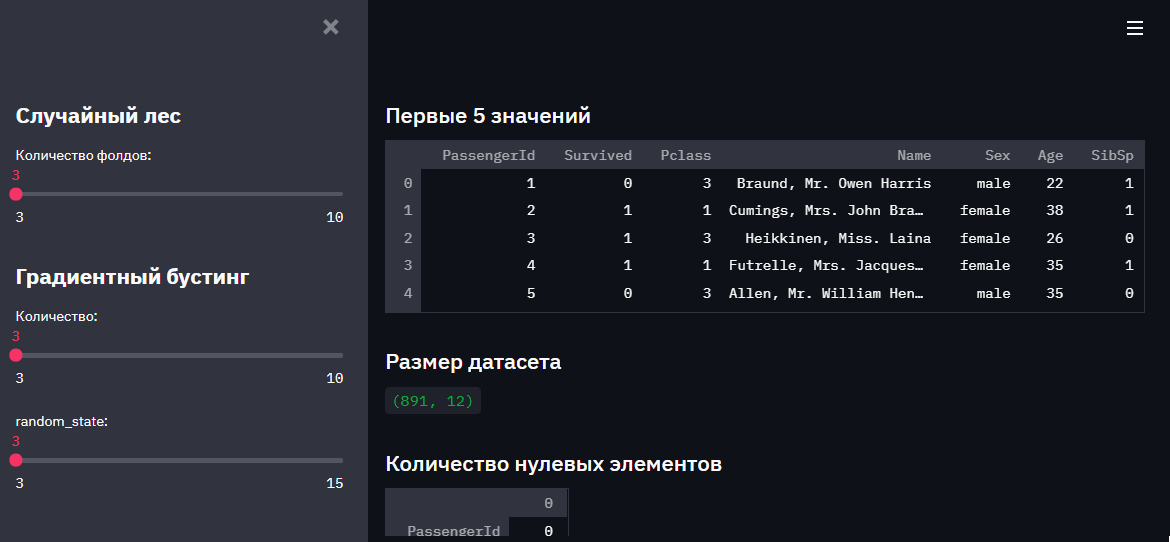


Рис.1 - ”*Основное окно проекта*”

Слева – меню ручной настройки двух из пяти моделей: Случайного леса и Градиентного бустинга (*см. Рис.1*).

Справа – информация об использующимся датасете: Первые 5 значений (*см. Рис.1*), Размер (*см. Рис.1*), Количество нулевых элементов(*см. Рис.2,3*), Колонки и их типы данных (*см. Рис.4*).

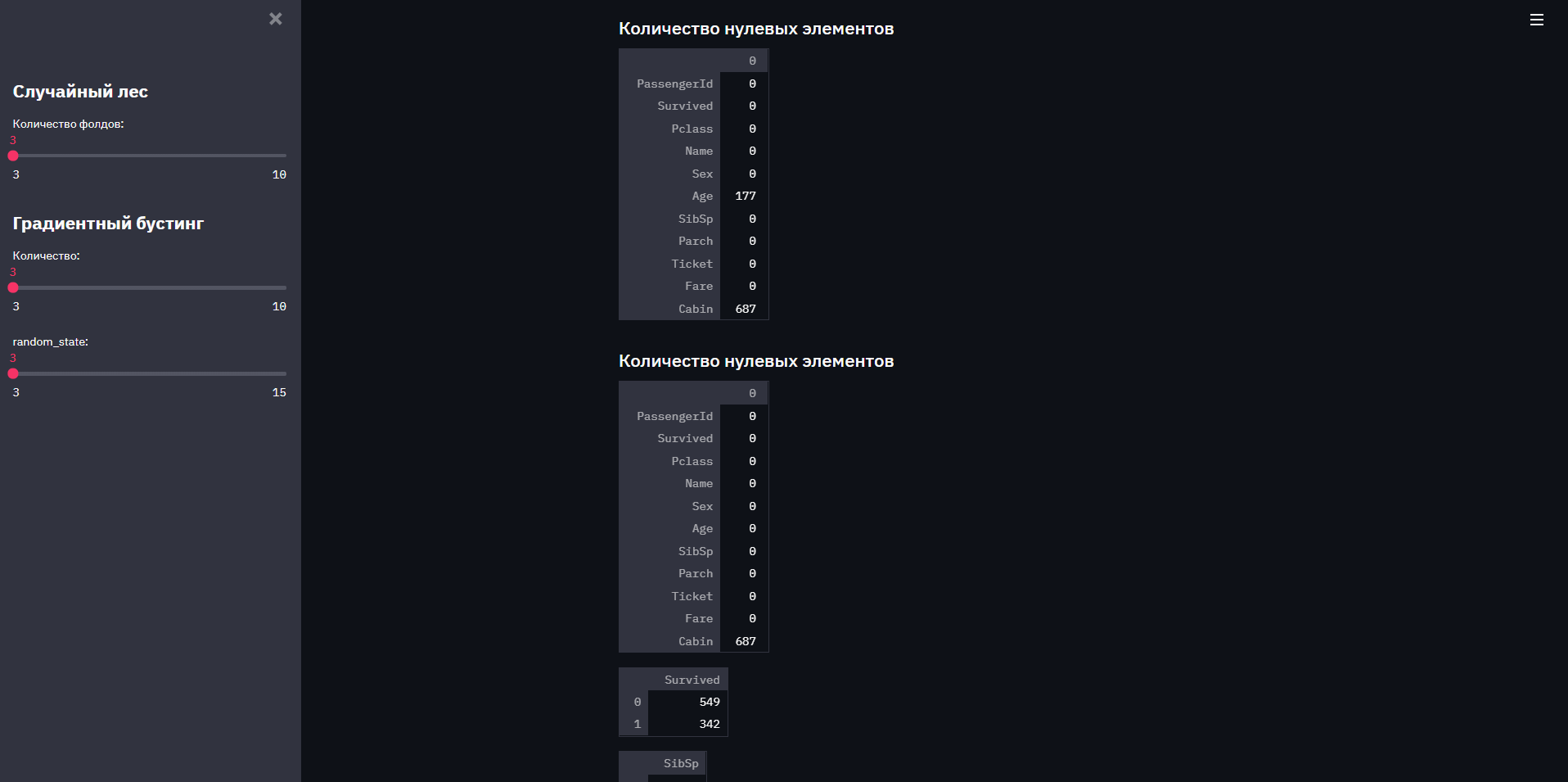


Рис.2 - ”*Количество нулевых элементов (1)*”

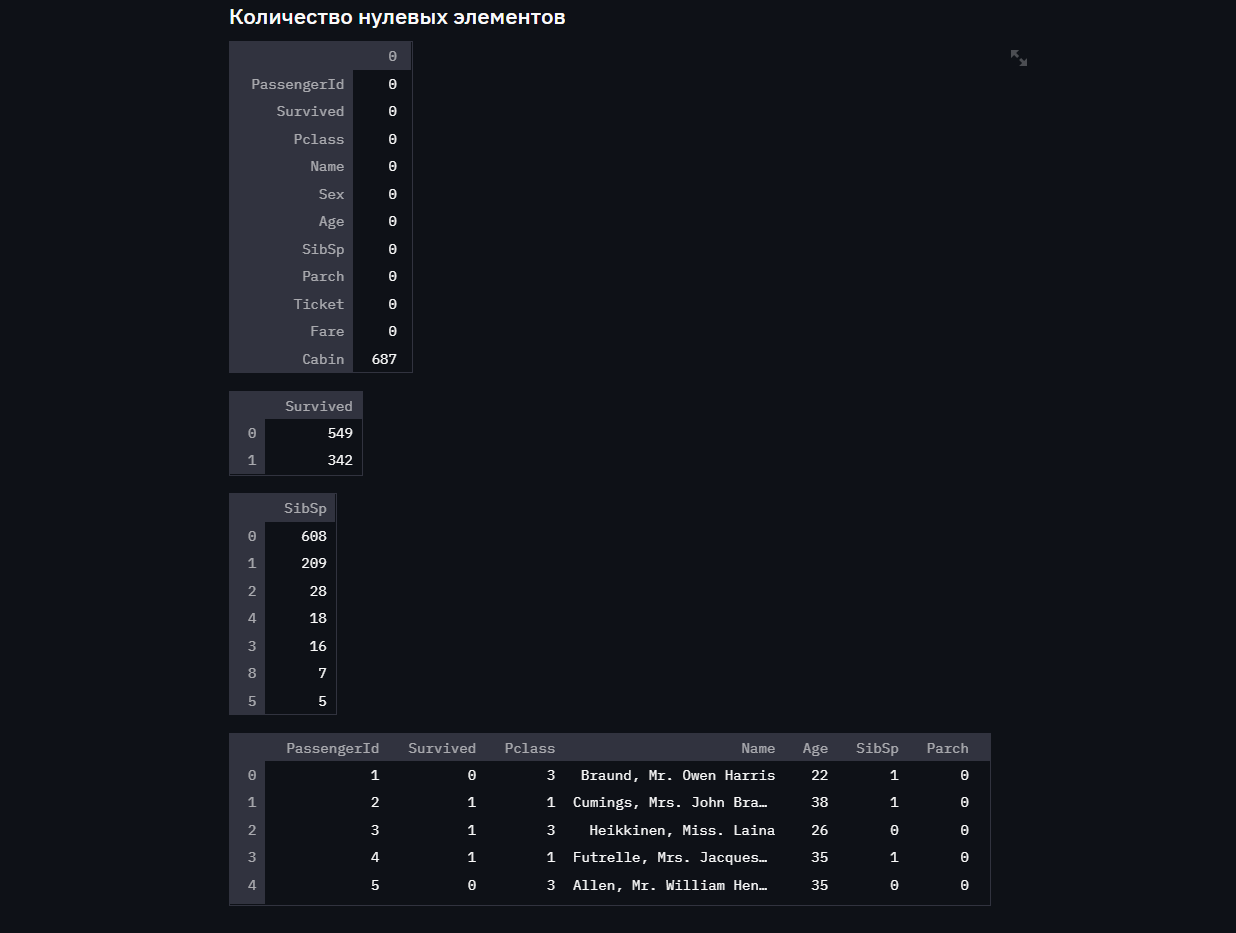


Рис.3 - ”*Количество нулевых элементов (2)*”

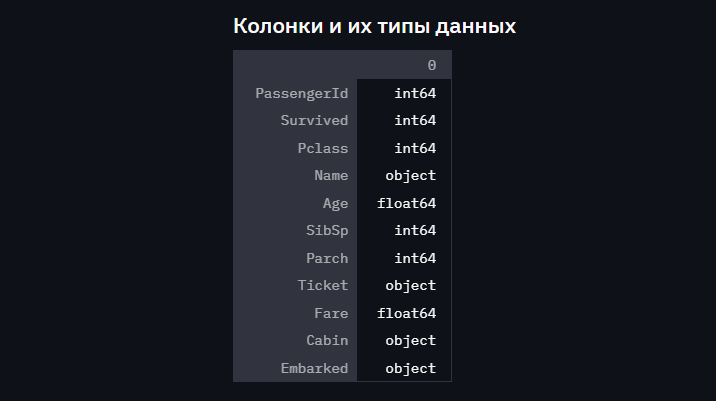


Рис.4 - ”*Колонки и их типы данных*”

2. Далее, вниз по странице, – статистические данные (см. Рис.5,6,7).

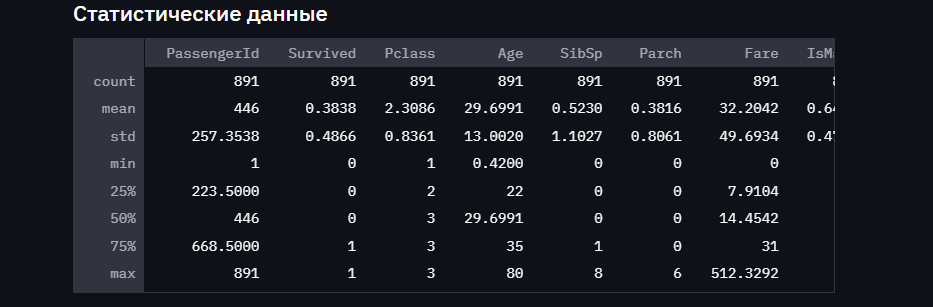


Рис.5 - ”*Статистические данные в форме таблицы*”

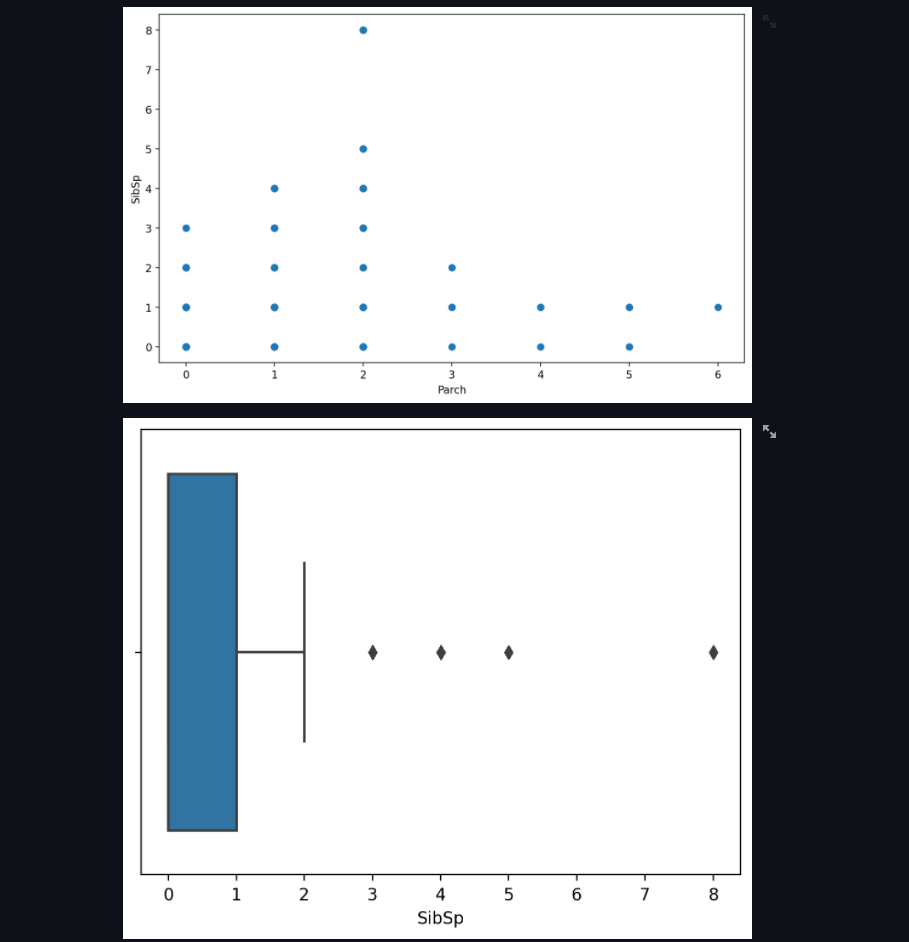
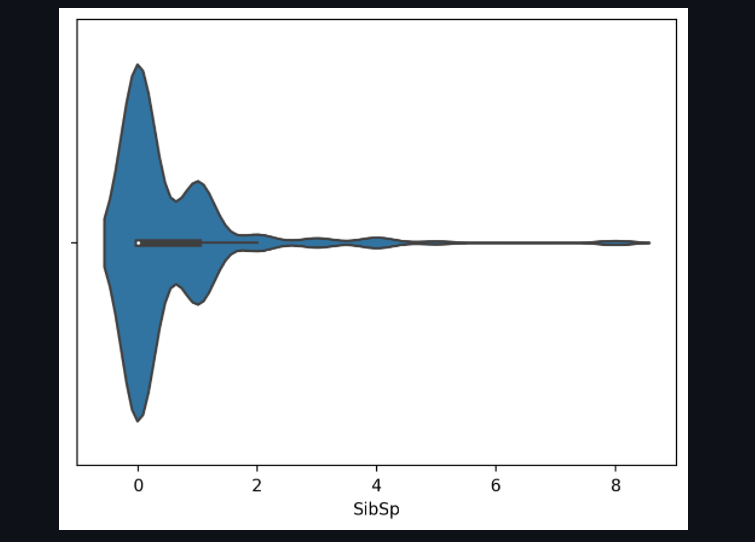


Рис.6 - ”*Статистические данные в графическом виде (1)*”

 Рис.7 - ”*Статистические данные в графическом виде (2)*”

3. Далее – идёт масштабирование данных (см. *Рис.8,9*).

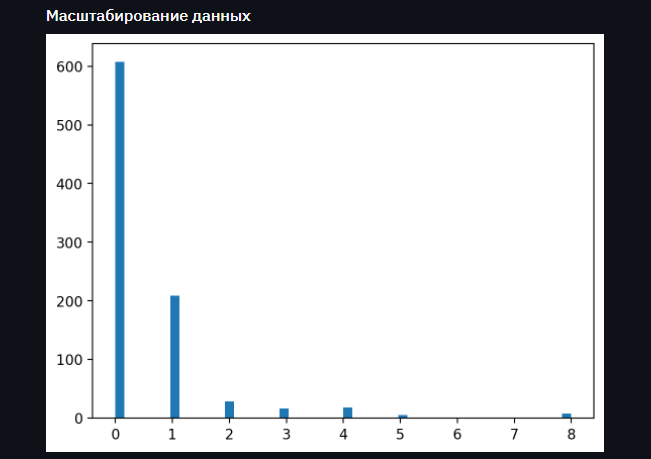


Рис.8 - ”*Масштабирование данных (1)* ”

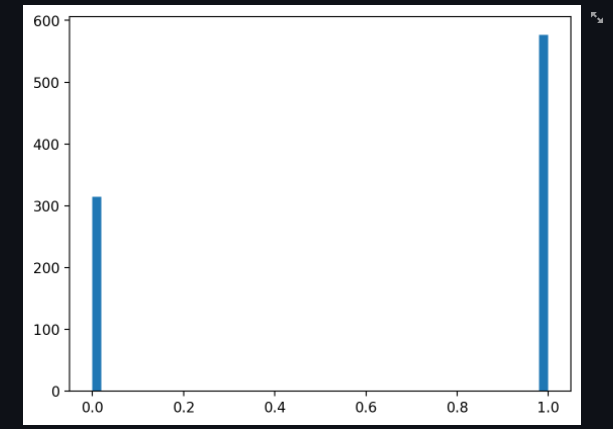
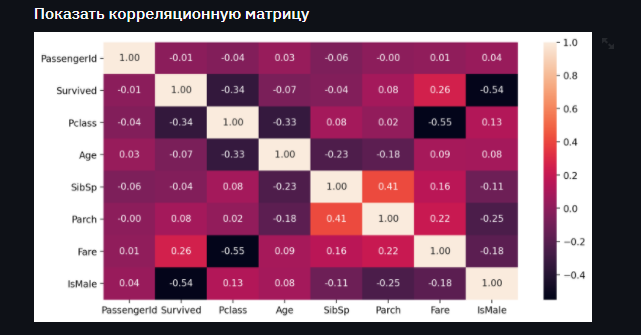


Рис.9 - ”*Масштабирование данных (2)*”

4. Далее – Корреляционная матрица (см. *Рис.10*).

 Рис.10 - ”*Корреляционная матрица*”

5. Далее – идёт ансамблевая модель ”Случайный лес” (см. Рис.11,12).

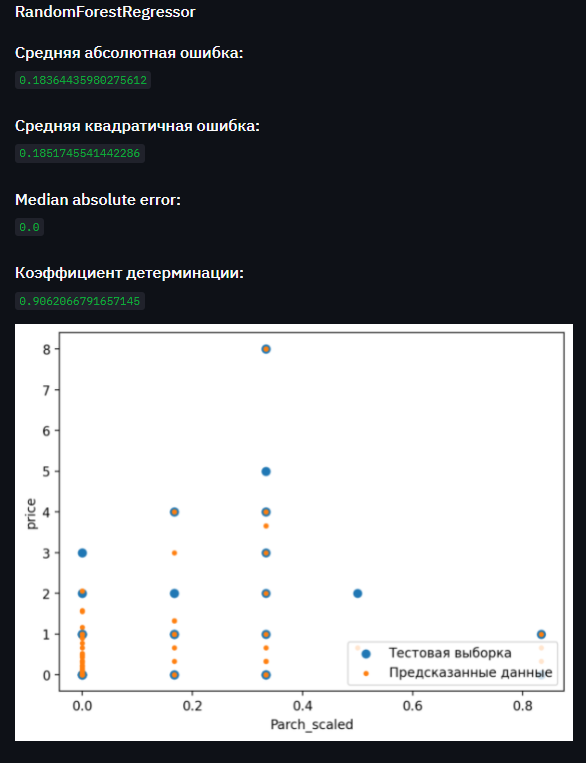


Рис.11 - ”*Случайный лес: построение модели по “Ручным данным“* ”

На рисунке 11 можно видеть модель, построенную по вручную настраиваемым данным, посредством ”меню слева”.

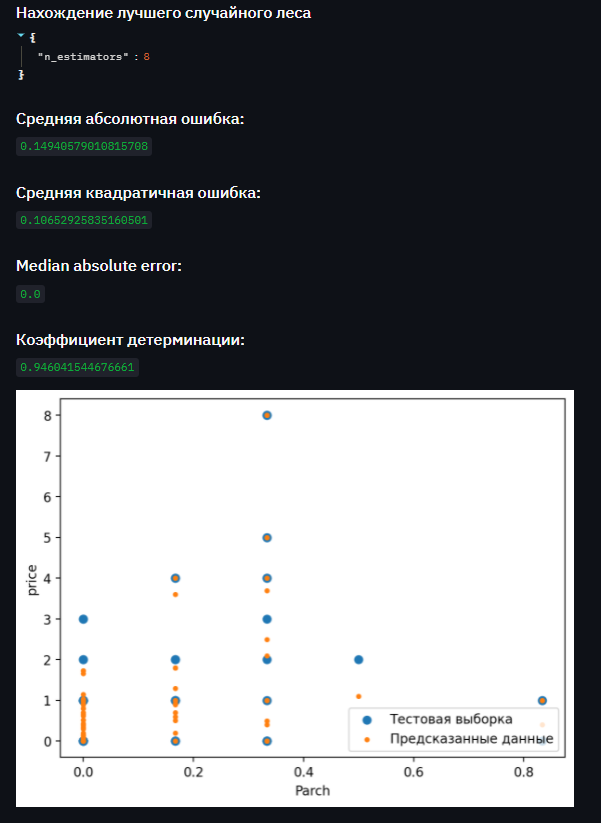


Рис.12 - ”*Случайный лес: нахождение лучшей модели*”

На рисунке 12 можно наблюдать поиск лучших параметров для модели, что, в результате, можно заметить по увеличившемуся коэффициенту детерминации.

6. Далее – идёт ансамблевая модель ”Градиентный бустинг” (см. Рис.13,14).

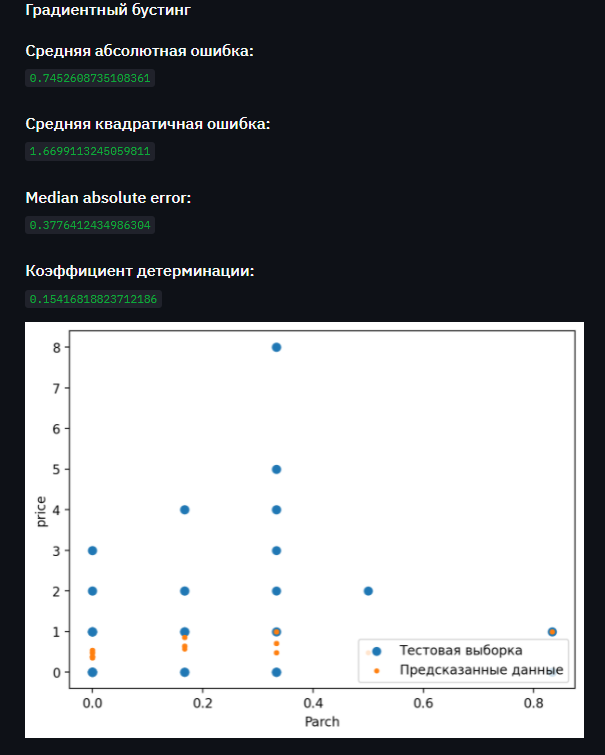


Рис.13 - ”*Градиентный бустинг: построение модели по “Ручным данным“* ”

На рисунке 13 можно видеть модель, построенную по вручную настраиваемым данным, посредством ”меню слева”.

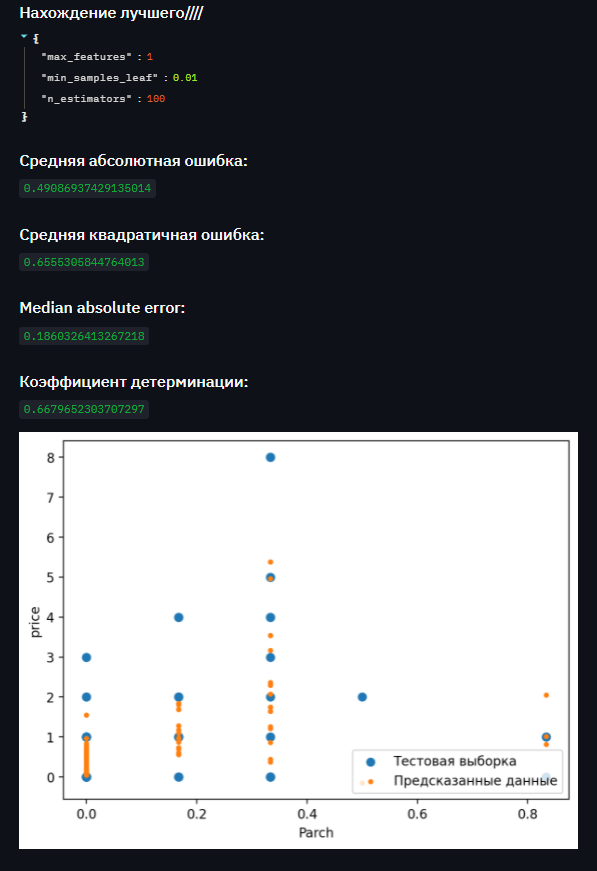


Рис.14 - ”*Градиентный бустинг: нахождение лучшей модели*”

На рисунке 14 можно наблюдать поиск лучших параметров для модели, что, в результате, можно заметить по увеличившемуся коэффициенту детерминации.

7. Далее – идёт построение модели линейной регрессии (см. Рис.15).

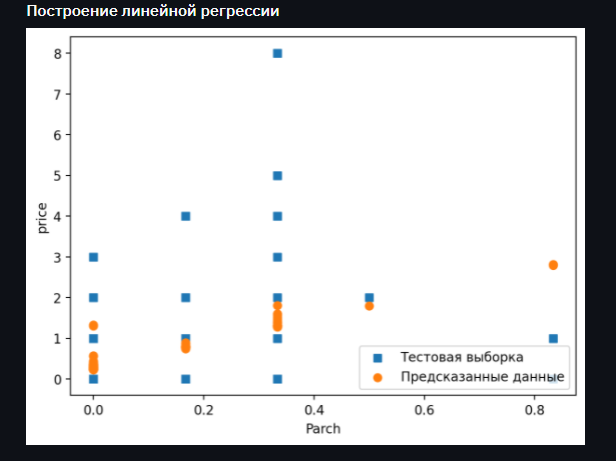
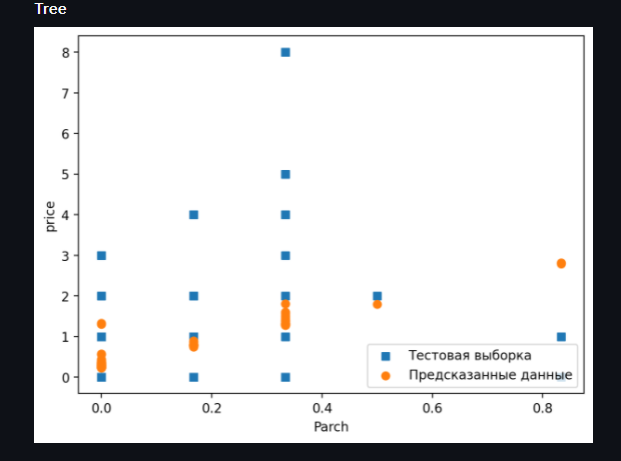
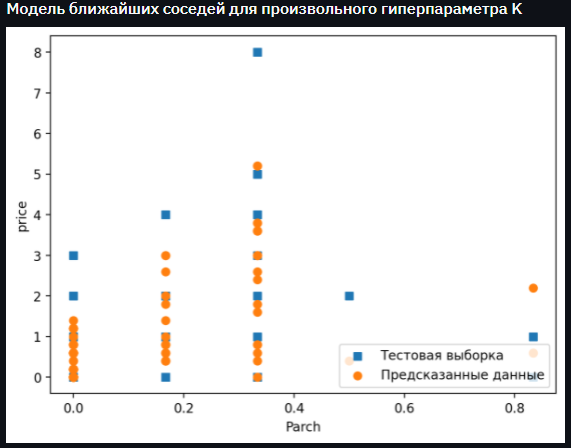


Рис.15 - ”*Линейная регрессия*”

8. Далее – идёт построение модели дерева (см. Рис.16).

 Рис.16 - ”*Дерево*”

9. Далее – идёт построение модели ближайших соседей для произвольного гиперпараметра K (см. Рис.17).

 Рис.17 - ”*Модель ближайших соседей для произвольного гиперпараметра*”

Код курсового проекта:

import streamlit as st

import seaborn as sns

import pandas as pd

import numpy as np

import plotly.figure\_factory as ff

from sklearn.preprocessing import MinMaxScaler

from sklearn.model\_selection import cross\_val\_score

from sklearn.model\_selection import GridSearchCV

from sklearn.neighbors import KNeighborsRegressor, KNeighborsClassifier

from sklearn.ensemble import RandomForestRegressor

import matplotlib.pyplot as plt

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

from sklearn.svm import SVC, LinearSVC

from sklearn.ensemble import GradientBoostingRegressor

from sklearn.preprocessing import StandardScaler, MinMaxScaler, StandardScaler, Normalizer

from sklearn.linear\_model import LinearRegression

from sklearn.metrics import mean\_absolute\_error, mean\_squared\_error, median\_absolute\_error, r2\_score

from sklearn.neighbors import KNeighborsRegressor

from sklearn import tree

import re

def load\_data():

    '''

    Загрузка данных

    '''

    data = pd.read\_csv('data/train.csv')

    return data

@st.cache

def preprocess\_data(data\_in):

    '''

    Масштабирование признаков, функция возвращает X и y для кросс-валидации

    '''

    data\_out = data\_in.copy()

    # Числовые колонки для масштабирования

    scale\_cols = ['Parch', 'Fare', 'IsMale']

    new\_cols = []

    sc1 = MinMaxScaler()

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

    for i in range(len(scale\_cols)):

        col = scale\_cols[i]

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

        new\_cols.append(new\_col\_name)

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

    X = data\_out[new\_cols]

    Y = data\_out['SibSp']

    # Чтобы в тесте получилось низкое качество используем только 0,5% данных для обучения

    X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, Y, train\_size=0.8, test\_size=0.2, random\_state=1)

    return X\_train, X\_test, y\_train, y\_test, X, Y

data = load\_data()

st.sidebar.header('Случайный лес')

n\_estimators\_1 = st.sidebar.slider('Количество фолдов:', min\_value=3, max\_value=10, value=3, step=1)

st.sidebar.header('Градиентный бустинг')

n\_estimators\_2 = st.sidebar.slider('Количество:', min\_value=3, max\_value=10, value=3, step=1)

random\_state\_2 = st.sidebar.slider('random\_state:', min\_value=3, max\_value=15, value=3, step=1)

# Первые пять строк датасета

st.subheader('Первые 5 значений')

st.write(data.head())

st.subheader('Размер датасета')

st.write(data.shape)

st.subheader('Количество нулевых элементов')

st.write(data.isnull().sum())

data['Age'] = data['Age'].replace(0,np.nan)

data['Age'] = data['Age'].fillna(data['Age'].mean())

st.subheader('Количество нулевых элементов')

st.write(data.isnull().sum())

st.write(data['Survived'].value\_counts())

st.write(data['SibSp'].value\_counts())

#  кодируем в 1/0

data['IsMale']=data.Sex.replace({'female':0,'male':1})

data.drop('Sex', axis = 1, inplace = True)

st.write(data.head())

st.subheader('Колонки и их типы данных')

st.write(data.dtypes)

st.subheader('Статистические данные')

st.write(data.describe())

fig, ax = plt.subplots(figsize=(10, 6))

ax.scatter(x=data['Parch'], y=data['SibSp'])

plt.xlabel("Parch")

plt.ylabel("SibSp")

st.pyplot(fig)

f1, ax = plt.subplots()

sns.boxplot(x=data['SibSp'])

st.pyplot(f1)

f, ax = plt.subplots()

sns.violinplot(x=data['SibSp'])

st.pyplot(f)

st.subheader('Масштабирование данных')

f, ax = plt.subplots()

plt.hist(data['SibSp'], 50)

plt.show()

st.pyplot(f)

f, ax = plt.subplots()

plt.hist(data['IsMale'], 50)

plt.show()

st.pyplot(f)

st.subheader('Показать корреляционную матрицу')

fig1, ax = plt.subplots(figsize=(10, 5))

sns.heatmap(data.corr(), annot=True, fmt='.2f')

st.pyplot(fig1)

X\_train, X\_test, Y\_train, Y\_test, X, Y = preprocess\_data(data)

forest\_1 = RandomForestRegressor(n\_estimators=n\_estimators\_1, oob\_score=True, random\_state=10)

forest\_1.fit(X, Y)

Y\_predict = forest\_1.predict(X\_test)

st.subheader('RandomForestRegressor')

st.subheader('Средняя абсолютная ошибка:')

st.write(mean\_absolute\_error(Y\_test, Y\_predict))

st.subheader('Средняя квадратичная ошибка:')

st.write(mean\_squared\_error(Y\_test, Y\_predict))

st.subheader('Median absolute error:')

st.write(median\_absolute\_error(Y\_test, Y\_predict))

st.subheader('Коэффициент детерминации:')

st.write(r2\_score(Y\_test, Y\_predict))

fig1 = plt.figure(figsize=(7, 5))

ax = plt.scatter(X\_test['Parch\_scaled'], Y\_test, marker='o', label='Тестовая выборка')

plt.scatter(X\_test['Parch\_scaled'], Y\_predict, marker='.', label='Предсказанные данные')

plt.legend(loc='lower right')

plt.xlabel('Parch\_scaled')

plt.ylabel('price')

plt.plot(n\_estimators\_1)

st.pyplot(fig1)

st.subheader('Нахождение лучшего случайного леса')

params2 = {

    'n\_estimators': [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 15, 20, 25, 50, 75, 100],

}

grid\_2 = GridSearchCV(estimator=RandomForestRegressor(oob\_score=True, random\_state=5),

                      param\_grid=params2,

                      scoring='neg\_mean\_squared\_error',

                      cv=3,

                      n\_jobs=-1)

grid\_2.fit(X, Y)

st.write(grid\_2.best\_params\_)

forest\_3 = RandomForestRegressor(n\_estimators=10, oob\_score=True, random\_state=5)

forest\_3.fit(X, Y)

Y\_predict3 = forest\_3.predict(X\_test)

st.subheader('Средняя абсолютная ошибка:')

st.write(mean\_absolute\_error(Y\_test, Y\_predict3))

st.subheader('Средняя квадратичная ошибка:')

st.write(mean\_squared\_error(Y\_test, Y\_predict3))

st.subheader('Median absolute error:')

st.write(median\_absolute\_error(Y\_test, Y\_predict3))

st.subheader('Коэффициент детерминации:')

st.write(r2\_score(Y\_test, Y\_predict3))

fig1 = plt.figure(figsize=(7, 5))

ax = plt.scatter(X\_test['Parch\_scaled'], Y\_test, marker='o', label='Тестовая выборка')

plt.scatter(X\_test['Parch\_scaled'], Y\_predict3, marker='.', label='Предсказанные данные')

plt.legend(loc='lower right')

plt.xlabel('Parch')

plt.ylabel('price')

plt.plot(n\_estimators\_1)

st.pyplot(fig1)

st.subheader('Градиентный бустинг')

grad = GradientBoostingRegressor(n\_estimators=n\_estimators\_2, random\_state=random\_state\_2)

grad.fit(X\_train, Y\_train)

Y\_grad\_pred = grad.predict(X\_test)

st.subheader('Средняя абсолютная ошибка:')

st.write(mean\_absolute\_error(Y\_test, Y\_grad\_pred))

st.subheader('Средняя квадратичная ошибка:')

st.write(mean\_squared\_error(Y\_test, Y\_grad\_pred))

st.subheader('Median absolute error:')

st.write(median\_absolute\_error(Y\_test, Y\_grad\_pred))

st.subheader('Коэффициент детерминации:')

st.write(r2\_score(Y\_test, Y\_grad\_pred))

fig2 = plt.figure(figsize=(7, 5))

ax = plt.scatter(X\_test['Parch\_scaled'], Y\_test, marker='o', label='Тестовая выборка')

plt.scatter(X\_test['Parch\_scaled'], Y\_grad\_pred, marker='.', label='Предсказанные данные')

plt.legend(loc='lower right')

plt.xlabel('Parch')

plt.ylabel('price')

plt.plot(random\_state\_2)

st.pyplot(fig2)

st.subheader('Нахождение лучшего////')

params = {

    'n\_estimators': [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 15, 20, 25, 50, 75, 100],

    'max\_features': [0.2, 0.3, 0.4, 0.6, 0.8, 0.9, 1.0],

    'min\_samples\_leaf': [0.01, 0.04, 0.06, 0.08, 0.1]

}

grid\_gr = GridSearchCV(estimator=GradientBoostingRegressor(random\_state=10),

                       param\_grid=params,

                       scoring='neg\_mean\_squared\_error',

                       cv=3,

                       n\_jobs=-1)

grid\_gr.fit(X\_train, Y\_train)

st.write(grid\_gr.best\_params\_)

grad1 = GradientBoostingRegressor(n\_estimators=100, max\_features=0.6, min\_samples\_leaf=0.01, random\_state=1)

grad1.fit(X\_train, Y\_train)

Y\_grad\_pred1 = grad1.predict(X\_test)

st.subheader('Средняя абсолютная ошибка:')

st.write(mean\_absolute\_error(Y\_test, Y\_grad\_pred1))

st.subheader('Средняя квадратичная ошибка:')

st.write(mean\_squared\_error(Y\_test, Y\_grad\_pred1))

st.subheader('Median absolute error:')

st.write(median\_absolute\_error(Y\_test, Y\_grad\_pred1))

st.subheader('Коэффициент детерминации:')

st.write(r2\_score(Y\_test, Y\_grad\_pred1))

fig1 = plt.figure(figsize=(7, 5))

ax = plt.scatter(X\_test['Parch\_scaled'], Y\_test, marker='o', label='Тестовая выборка')

plt.scatter(X\_test['Parch\_scaled'], Y\_grad\_pred1, marker='.', label='Предсказанные данные')

plt.legend(loc='lower right')

plt.xlabel('Parch')

plt.ylabel('price')

plt.plot(n\_estimators\_1)

st.pyplot(fig1)

st.subheader('Построение линейной регрессии')

Lin\_Reg = LinearRegression().fit(X\_train, Y\_train)

lr\_y\_pred = Lin\_Reg.predict(X\_test)

fig3 = plt.figure(figsize=(7, 5))

plt.scatter(X\_test['Parch\_scaled'], Y\_test, marker='s', label='Тестовая выборка')

plt.scatter(X\_test['Parch\_scaled'], lr\_y\_pred, marker='o', label='Предсказанные данные')

plt.legend(loc='lower right')

plt.xlabel('Parch')

plt.ylabel('price')

plt.show()

st.pyplot(fig3)

st.subheader('Tree')

clf = tree.DecisionTreeClassifier()

clf = clf.fit(X\_train, Y\_train)

lr\_y\_pred = Lin\_Reg.predict(X\_test)

fig5 = plt.figure(figsize=(7, 5))

plt.scatter(X\_test['Parch\_scaled'], Y\_test, marker='s', label='Тестовая выборка')

plt.scatter(X\_test['Parch\_scaled'], lr\_y\_pred, marker='o', label='Предсказанные данные')

plt.legend(loc='lower right')

plt.xlabel('Parch')

plt.ylabel('price')

plt.show()

st.pyplot(fig5)

st.subheader('Модель ближайших соседей для произвольного гиперпараметра K')

Regressor\_5NN = KNeighborsRegressor(n\_neighbors = 5)

Regressor\_5NN.fit(X\_train, Y\_train)

lr\_y\_pred = Regressor\_5NN.predict(X\_test)

fig6 = plt.figure(figsize=(7, 5))

plt.scatter(X\_test['Parch\_scaled'], Y\_test, marker='s', label='Тестовая выборка')

plt.scatter(X\_test['Parch\_scaled'], lr\_y\_pred, marker='o', label='Предсказанные данные')

plt.legend(loc='lower right')

plt.xlabel('Parch')

plt.ylabel('price')

plt.show()

st.pyplot(fig6)