



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 _____

КАФЕДРА _____

РАСЧЕТНО-ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

К КУРСОВОМУ ПРОЕКТУ

НА ТЕМУ:

Прогнозирование отношения студентов к
дистанционному обучению с применением моделей
машинного обучения

Студент ИУ5-34М
(Группа)

_____ Арбузов А.П.
(Подпись, дата) (И.О.Фамилия)

Руководитель курсового проекта

_____ 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)
« _____ » 20 ____ г.

З А Д А Н И Е на выполнение курсового проекта

по дисциплине Обработка и анализ данных

Студент группы ИУ5-34М

Арбузов Алексей Павлович
(Фамилия, имя, отчество)

Тема курсового проекта: Прогнозирование отношения студентов к дистанционному обучению с применением моделей машинного обучения

Направленность КП (учебный, исследовательский, практический, производственный, др.)
учебная

Источник тематики (кафедра, предприятие, НИР) кафедра

График выполнения проекта: 25% к ____ нед., 50% к ____ нед., 75% к ____ нед., 100% к ____ нед.

Задание

Оформление курсового проекта:

Расчёто-пояснительная записка на ____ листах формата А4.

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

Дата выдачи задания « ____ » 2021 г.

Руководитель курсового проекта

(Подпись, дата) Ю.Е. Гапанюк
(И.О.Фамилия)

Студент

(Подпись, дата) А.П. Арбузов
(И.О.Фамилия)

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

Введение:

Переход на дистанционный режим из-за угрозы заражения вирусной инфекцией сильно повлиял на процесс обучения студентов в разных странах. Для оценки удовлетворённости студентов новым способом обучения был проведён онлайн опрос, результаты которого приведены в датасете Online Survey Data. Ниже представлена его структура:

Имя поля	Значение поля
Level	Уровень образования
Age	Возраст
devicesUsage	Опыт использования электронных устройств до дистанционного обучения
resultIncreased	Улучшение результата (баллов) после перехода на дистанционное обучение
knowledgeIncreased	Улучшение знаний после перехода на дистанционное обучение
target	Целевой признак
educationArea	Где обучался студент (Город, Сельская местность)
internet	Был ли у студента доступ в интернет
internetType	Тип интернета (мобильный, проводной)
hoursBefore	Время на обучение до перехода на дистанционный режим
hoursAfter	Время на обучение после перехода на дистанционный режим
performanceIncreased	Улучшение успеваемости после перехода на дистанционное обучение
instituteType	Тип института (закрытый, открытый)
location	Текущее местоположение (Город, Сельская местность)
Gender	Пол
Issues	Были ли проблемы при дистанционном обучении
device	Устройство, через которое обучался студент (телефон, компьютер)

Выполнение:

```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt

from sklearn.preprocessing import LabelEncoder
from sklearn.impute import KNNImputer

from sklearn.model_selection import train_test_split
from sklearn.feature_selection import mutual_info_classif, chi2, f_classif

from sklearn.neighbors import KNeighborsClassifier
from sklearn.linear_model import LogisticRegression
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import RandomForestClassifier
from sklearn.ensemble import GradientBoostingClassifier

from sklearn.metrics import accuracy_score, precision_score, recall_score

import warnings
warnings.filterwarnings('ignore')
sns.set_style('whitegrid')
```

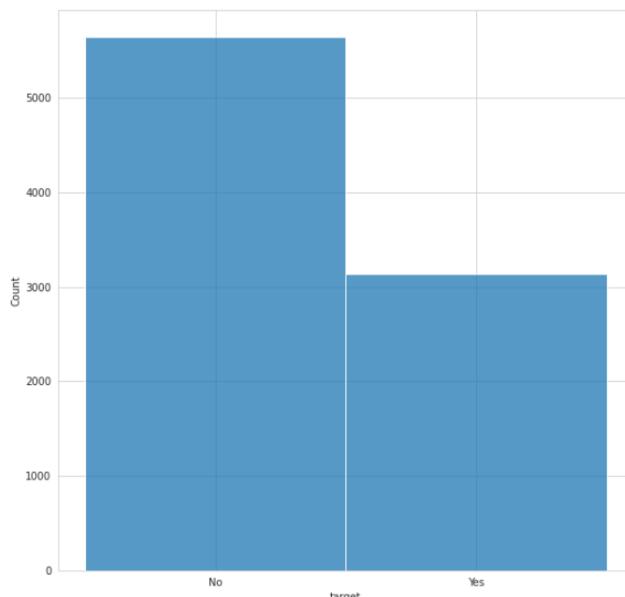
	level	age	devicesUsage	resultIncreased	knowledgeIncreased	target	educationArea	internet	internetType	hoursBefore	hoursAfter	performanceIncrea	
0	Upto HSC	20.0	Yes	No		Yes	No	Urban	No	Broadband	4	3	
1	Hons or Grater	25.0		No	No		No	No	Urban	No	Mobile Internet	4	4
2	Hons or Grater	25.0		Yes	Yes		Yes	Yes	Rural	No	Mobile Internet	5	2
3	Upto HSC	21.0		Yes	Yes		No	Yes	Urban	Yes	Mobile Internet	5	3
4	Hons or Grater	22.0		Yes	No		No	No	Rural	No	Mobile Internet	4	2

```
data.shape
```

```
(8783, 17)
```

Размер датасета составляет 8783 записи до обработки пустых полей. Рассмотрим количественное отношение целевого класса.

```
fig, ax = plt.subplots(figsize=(10,10))
sns.histplot(data['target'], discrete=True)
<AxesSubplot:xlabel='target', ylabel='Count'>
```



Как видно из гистограммы, число студентов, недовольных онлайн обучением больше.

Обработка пропущенных значений

```
print('Процент пустых данных по столбцам: ')
emptyMap = dict(zip(data.columns, data.isnull().sum()))
emptyArr = []

for col, empty in emptyMap.items():
    if empty != 0:
        print('{0}\t{1}'.format(round(empty / data.shape[0] * 100, 2), col))
        emptyArr.append(col)
```

Процент пустых данных по столбцам:

5.07	age
2.14	devicesUsage
3.68	resultIncreased
6.02	educationArea
8.27	instituteType
8.27	location
7.7	Gender
7.98	Issues

В датасете присутствуют пустые поля, которые необходимо обработать.

Процент всех пустых полей невелик, поэтому просто удалим записи с пустыми полями.

```
for col in emptyArr:
    data = data.dropna(subset=[col])
data.head()
```

	level	age	devicesUsage	resultIncreased	knowledgeIncreased	target	educationArea	internet	internetType	hoursBefore	hoursAfter	performanceIncreased
0	Upto HSC	20.0		Yes	No		Yes	No	Urban	No	Broadband	4
1	Hons or Grater	25.0		No	No		No	No	Urban	No	Mobile Internet	4
2	Hons or Grater	25.0		Yes	Yes		Yes	Yes	Rural	No	Mobile Internet	5
3	Upto HSC	21.0		Yes	Yes		No	Yes	Urban	Yes	Mobile Internet	5
4	Hons or Grater	22.0		Yes	No		No	No	Rural	No	Mobile Internet	4

Кодирование категориальных признаков

```
data.dtypes
```

level	object
age	float64
devicesUsage	object
resultIncreased	object
knowledgeIncreased	object
target	object
educationArea	object
internet	object
internetType	object
hoursBefore	int64
hoursAfter	int64
performanceIncreased	object
instituteType	object
location	object
Gender	object
Issues	object
device	object
dtype: object	

```
def fitTransrotmRepr(df, col):
    encoder = LabelEncoder()
    df[[col]] = encoder.fit_transform(df[[col]])
    encoders = df[col].unique()
    labels = encoder.inverse_transform(encoders)
    labelMap = dict(zip(labels, encoders))
    print('{0}: {1}'.format(col, labelMap))
```

В датасете почти все поля не являются числовыми значениями. Введём функцию для преобразования их в числовой тип через кодирование индексами. В процессе работы функция выведет на экран связку индекса и значения.

```

typeMap = dict(zip(data.columns, data.dtypes))

for col, typ in typeMap.items():
    if(typ == 'object'):
        fitTransrotmRepr(data, col)

data.head()

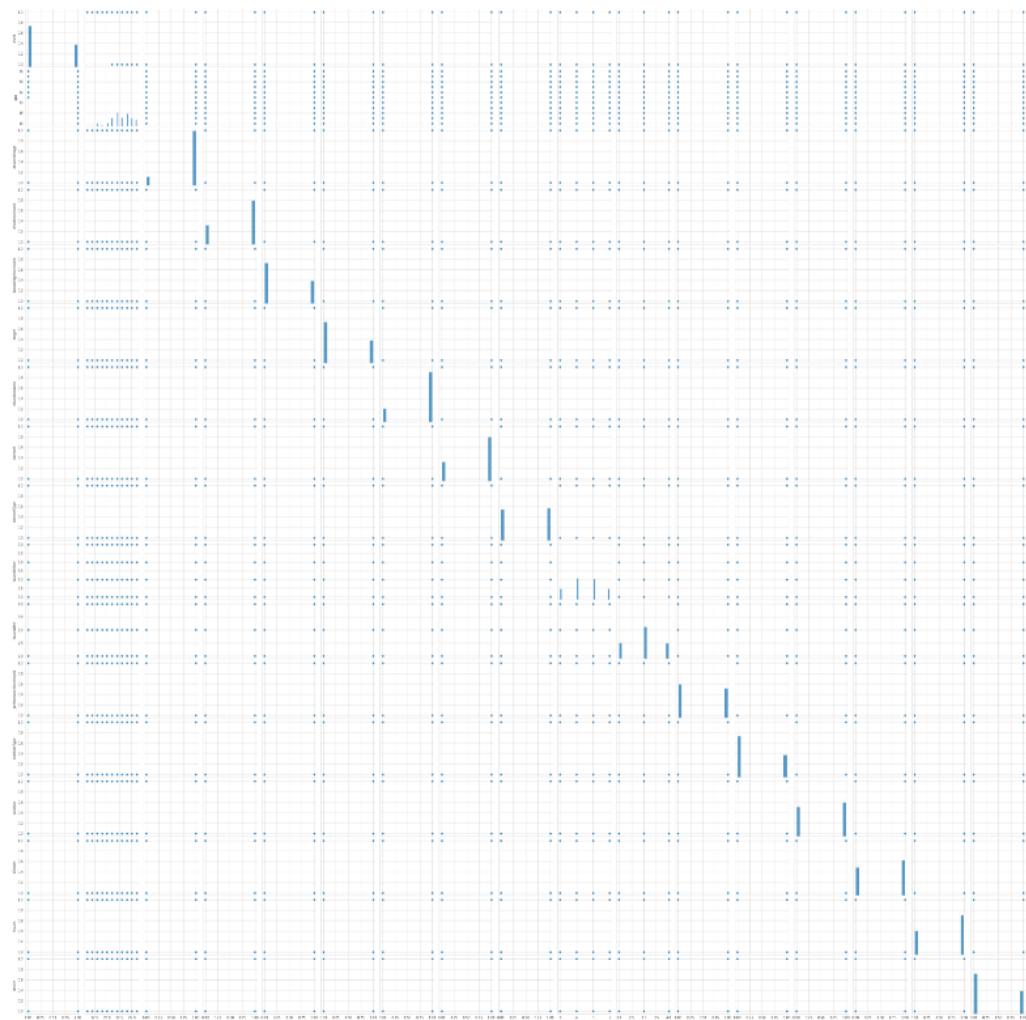
level: {'Upto HSC': 1, 'Hons or Grater': 0}
devicesUsage: {'Yes': 1, 'No': 0}
resultIncreased: {'No': 0, 'Yes': 1}
knowledgeIncreased: {'Yes': 1, 'No': 0}
target: {'No': 0, 'Yes': 1}
educationArea: {'Urban': 1, 'Rural': 0}
internet: {'No': 0, 'Yes': 1}
internetType: {'Broadband': 0, 'Mobile Internet': 1}
performanceIncreased: {'No': 0, 'Yes': 1}
instituteType: {'Public': 1, 'Private': 0}
location: {'Rural': 0, 'Urban': 1}
Gender: {'Male': 1, 'Female': 0}
Issues: {'Yes': 1, 'No': 0}
device: {'Mobile': 1, 'Computer': 0}

```

	level	age	devicesUsage	resultIncreased	knowledgeIncreased	target	educationArea	internet	internetType	hoursBefore	hoursAfter	performanceIncreased
0	1	20.0		1	0		1	0		0	4	3
1	0	25.0		0	0		0	0		1	4	4
2	0	25.0		1	1		1	1		0	0	1
3	1	21.0		1	1		0	1		1	1	5
4	0	22.0		1	0		0	0		0	0	1

Просмотр полученных данных

```
sns.pairplot(data)
```

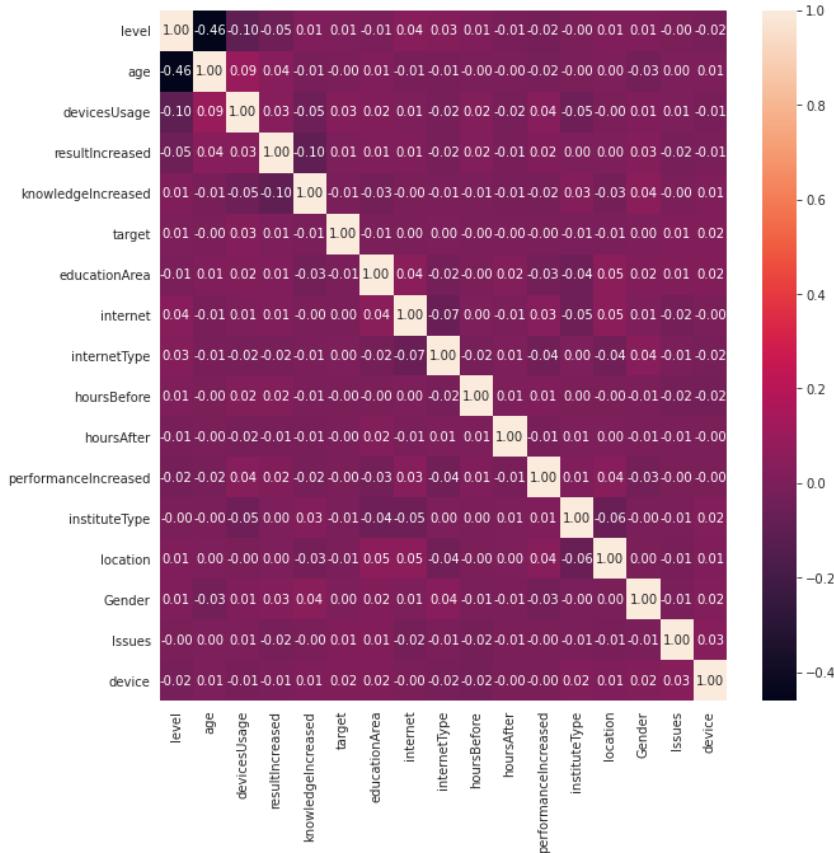


На диаграммах выше построены диаграммы распределений и зависимостей полей датасета.

Отбор признаков

Отбор на основе корреляции

```
fig, ax = plt.subplots(figsize=(10,10))
sns.heatmap(data.corr(method='pearson'), annot=True, fmt='.2f')
<AxesSubplot:>
```



Из Корреляционной матрицы видно, что нет зависимостей между целевым полем и остальными полями.

Отбор методом на основе статистических характеристик и методом вложений

```
class FeatureSelector:
    def __init__(self, df, target, method, index, isModel):
        if isModel:
            model = method()
            model.fit(df.drop(columns=[target]), df[target])
            self.mic = model.feature_importances_

        elif index:
            self.mic = method(df.drop(columns=[target]), df[target])[1]

        else:
            self.mic = method(df.drop(columns=[target]), df[target])

        self.mic = pd.Series(self.mic)
        self.mic.index = df.drop(columns=[target]).columns

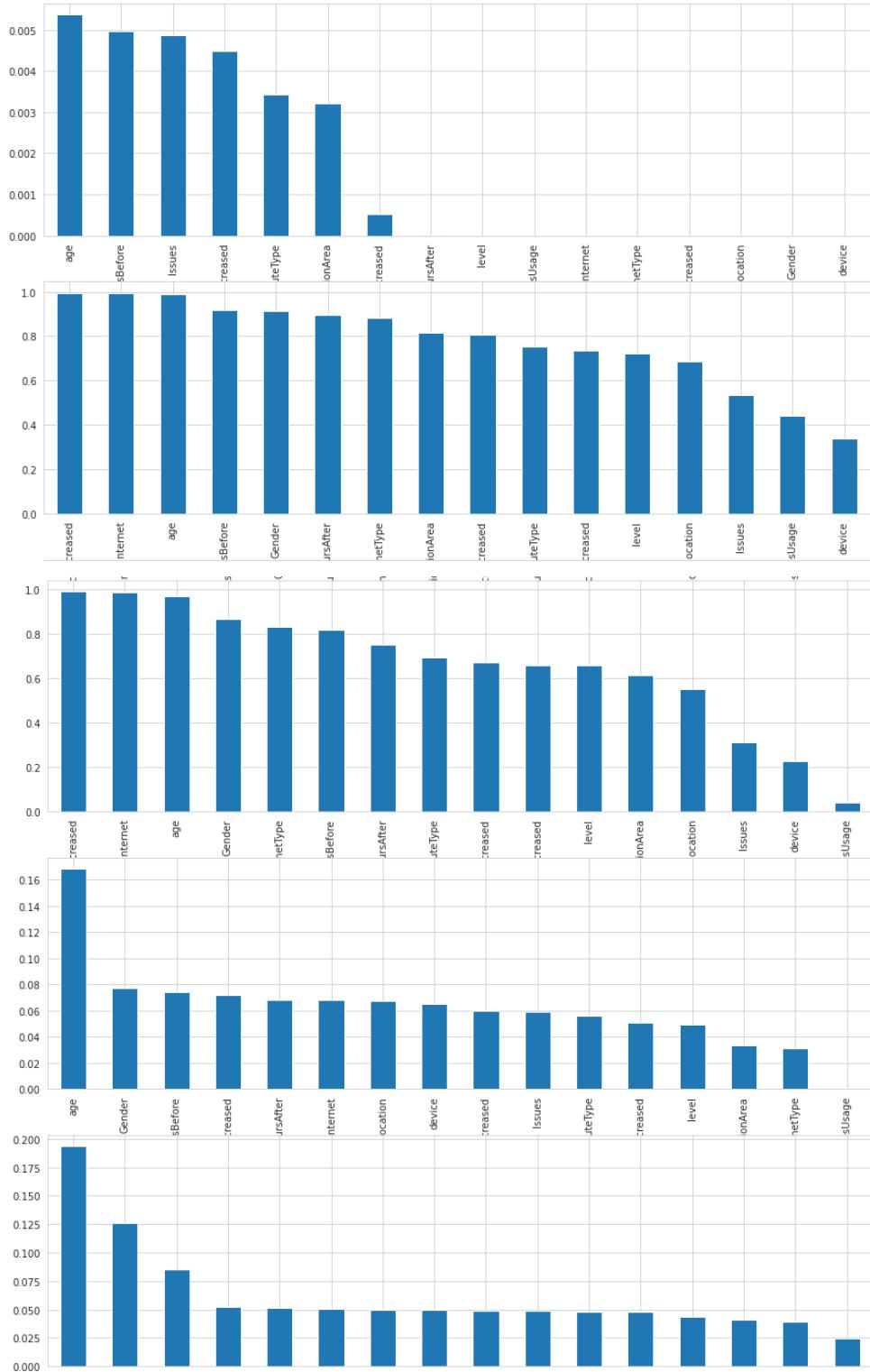
    def represent(self):
        self.mic.sort_values(ascending=False).plot.bar()
```

Введём класс для вывода статистики о важности полей с точки зрения различных методов их отбора.

```

selectors = [
    FeatureSelector(data, 'Survived', mutual_info_classif, False, False),
    FeatureSelector(data, 'Survived', chi2, True, False),
    FeatureSelector(data, 'Survived', f_classif, True, False),
    FeatureSelector(data, 'Survived', DecisionTreeClassifier, False, True),
    FeatureSelector(data, 'Survived', RandomForestClassifier, False, True)
]
plt.subplots(figsize=(15,25))
for i in range(len(selectors)):
    plt.subplot(5, 1, i + 1)
    selectors[i].represent()

```



Из полученных диаграмм видно, что важными признаками оказались почти все поля, за исключением: Проблемы, Устройство, Использование устройства до ДУ. Рассмотрим как важные признаки распределены эти поля по отношению к целевому признаку.

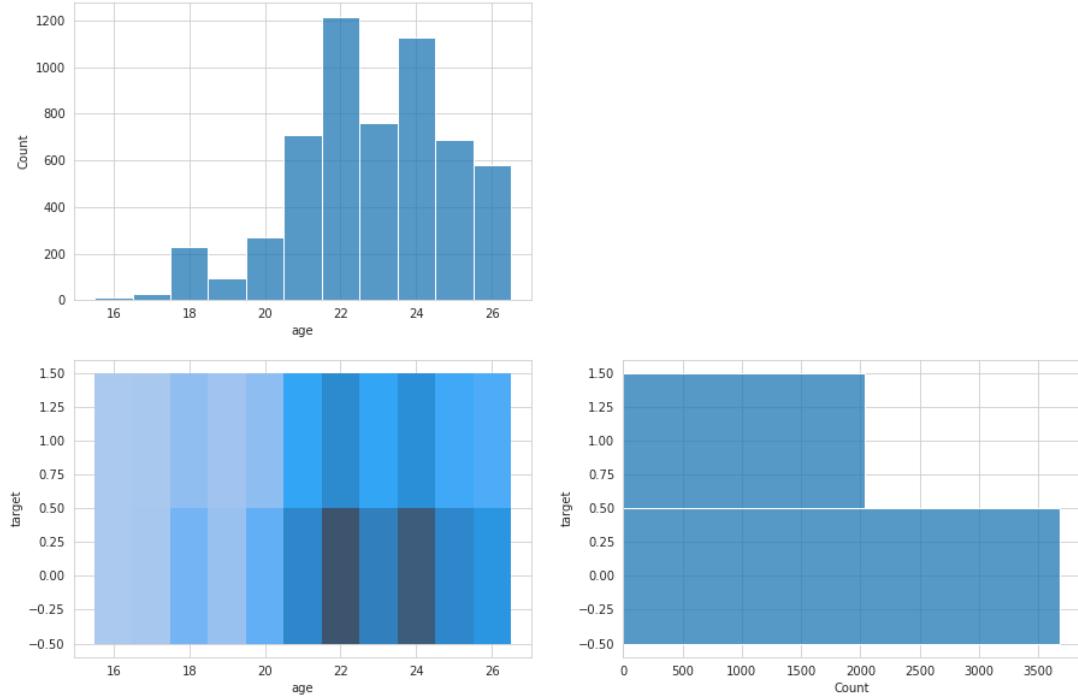
```
def joinhist(df, col, target):
    plt.subplots(figsize=(15,10))

    plt.subplot(2, 2, 1)
    sns.histplot(df[col], discrete=True)

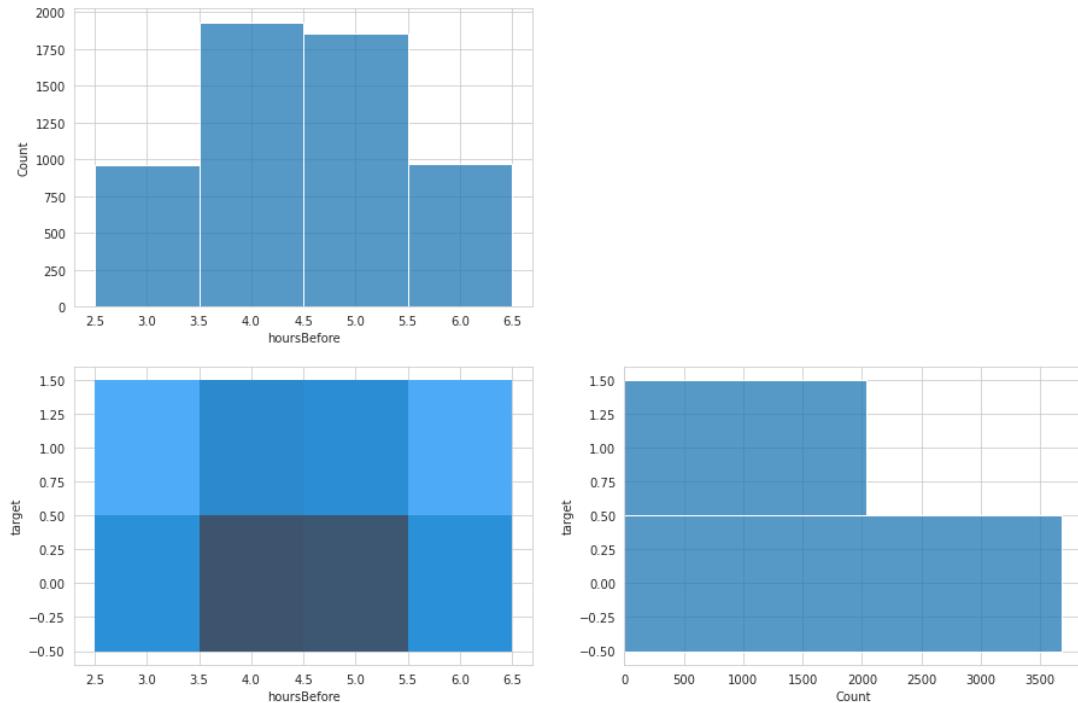
    plt.subplot(2, 2, 3)
    sns.histplot(df, x=col, y=target, discrete=True)

    plt.subplot(2, 2, 4)
    sns.histplot(y = df[target], discrete=True)

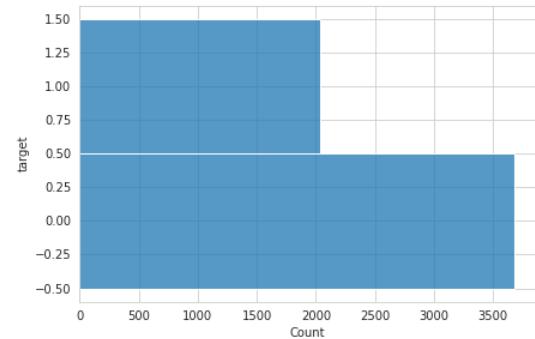
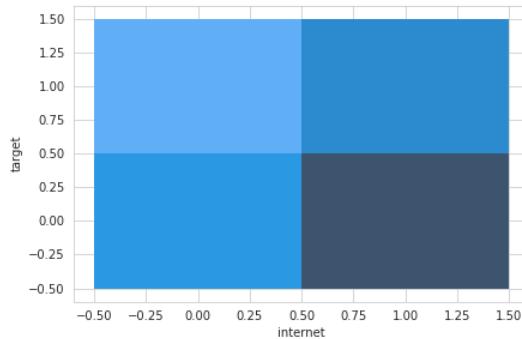
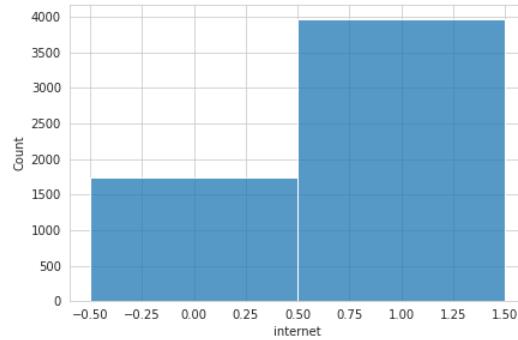
joinhist(data, 'age', 'target')
```



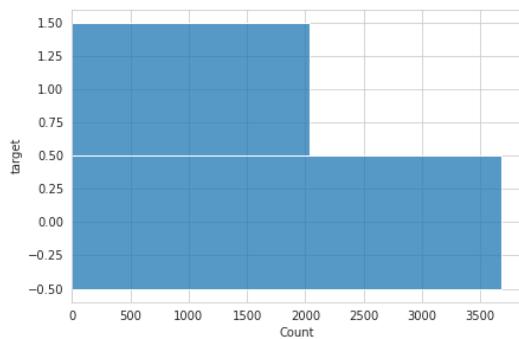
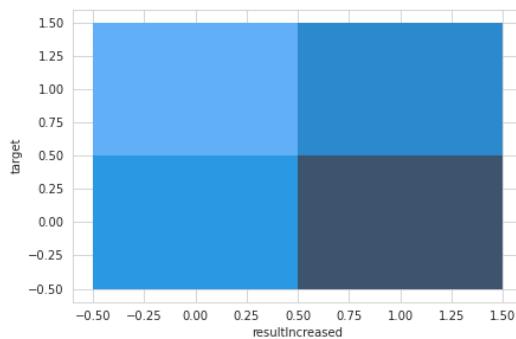
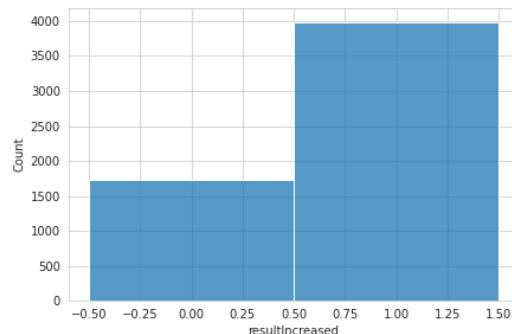
```
joinhist(data, 'hoursBefore', 'target')
```



```
joinhist(data, 'internet', 'target')
```

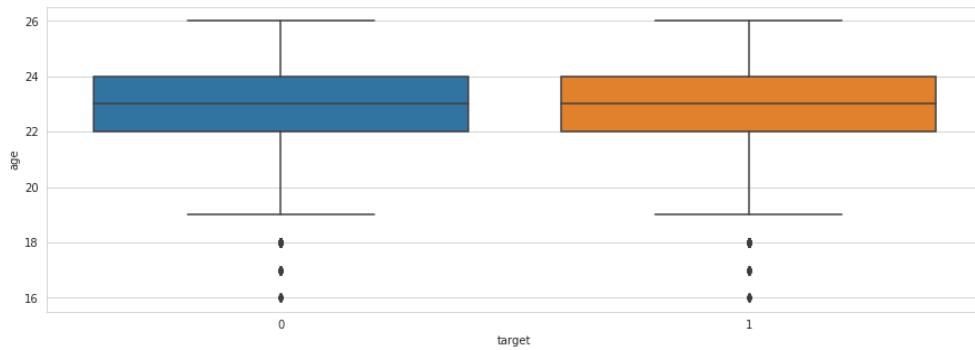


```
joinhist(data, 'resultIncreased', 'target')
```



```
fig, ax = plt.subplots(figsize=(15,5))
sns.boxplot(x='target', y='age', data=data)
```

```
<AxesSubplot:xlabel='target', ylabel='age'>
```



```

selectedFeatures = [
    'knowledgeIncreased',
    'resultIncreased',
    'performanceIncreased',
    'level',
    'age',
    'educationArea',
    'internet',
    'internetType',
    'hoursBefore',
    'hoursAfter',
    'instituteType',
    'location',
    'Gender'
]

```

После отбора полей можно приступить к обучению моделей.

Обучение моделей

Обучим один набор моделей дважды, один раз на всех полях датасета, второй раз на отобранных.

```

models = {
    'KNN': KNeighborsClassifier,
    'LR': LogisticRegression,
    'DTC': DecisionTreeClassifier,
    'RFC': RandomForestClassifier,
    'GBC': GradientBoostingClassifier
}

```

```

xTrain, xTest, yTrain, yTest = train_test_split(
    data.drop(columns=['target']),
    data['target'], test_size=0.3,
    random_state = 1)

```

```

report = []
for modelName, model in models.items():
    tmp = [modelName + '_all',]
    currModel = model()
    currModel.fit(xTrain, yTrain)
    yPred = currModel.predict(xTest)
    tmp.append(accuracy_score(yTest, yPred))
    tmp.append(precision_score(yTest, yPred))
    tmp.append(recall_score(yTest, yPred))
    report.append(tmp)

```

```

xTrain, xTest, yTrain, yTest = train_test_split(
    data[selectedFeatures],
    data['target'],
    test_size=0.3,
    random_state = 1)

```

```

for modelName, model in models.items():
    tmp = [modelName + '_sel',]
    currModel = model()
    currModel.fit(xTrain, yTrain)
    yPred = currModel.predict(xTest)
    tmp.append(accuracy_score(yTest, yPred))
    tmp.append(precision_score(yTest, yPred))
    tmp.append(recall_score(yTest, yPred))
    report.append(tmp)

```

```

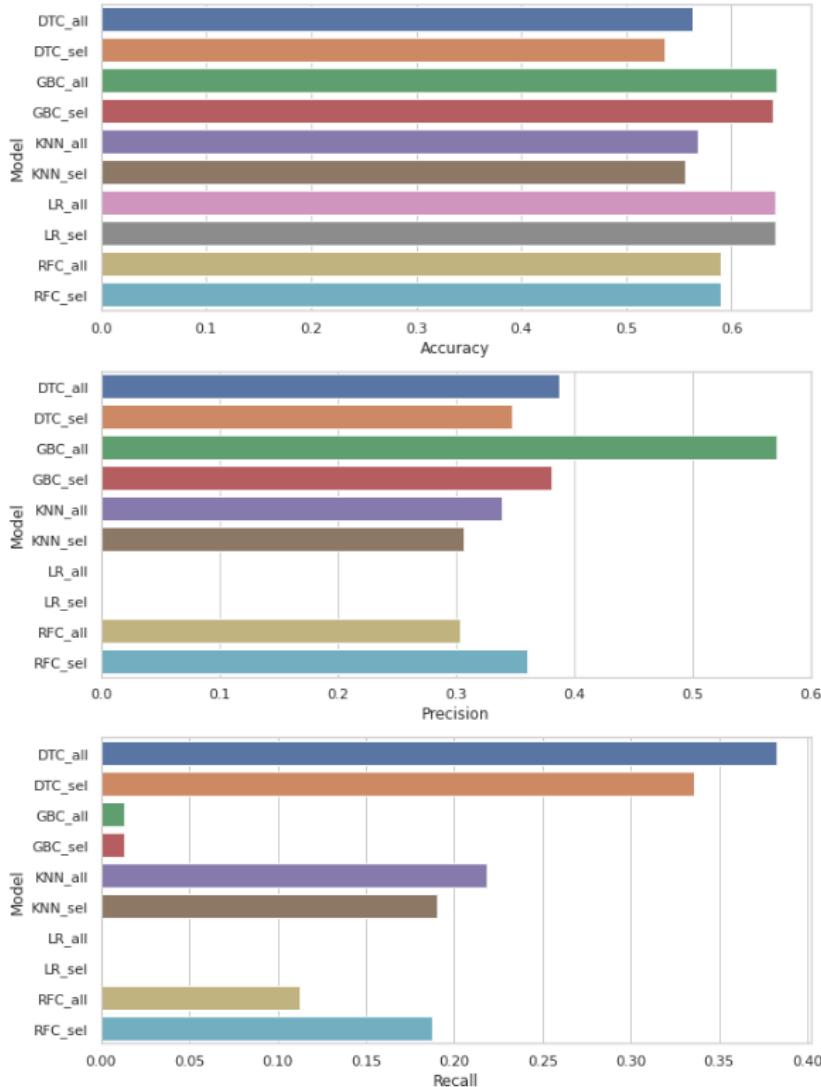
dfReport = pd.DataFrame(report, columns=['Model','Accuracy','Precision','Recall'])
dfReport = dfReport.sort_values(by=['Model'])
dfReport.head(10)

```

	Model	Accuracy	Precision	Recall
2	DTC_all	0.562682	0.387789	0.382736
7	DTC_sel	0.536443	0.347386	0.335505
4	GBC_all	0.643149	0.571429	0.013029
9	GBC_sel	0.639067	0.380952	0.013029
0	KNN_all	0.567930	0.339241	0.218241
5	KNN_sel	0.555685	0.306283	0.190554
1	LR_all	0.641983	0.000000	0.000000
6	LR_sel	0.641983	0.000000	0.000000
3	RFC_all	0.590087	0.303965	0.112378
8	RFC_sel	0.590087	0.360502	0.187296

Полученные данные с метриками выведем в виде диаграмм для наглядности результатов:

```
i = 1
for col in dfReport.drop(columns=['Model']):
    sns.set(rc={'figure.figsize':(15,15)})
    sns.set_style("whitegrid")
    plt.subplot(3, 1, i)
    sns.barplot(x=col, y="Model", data=dfReport)
    i += 1
```



Выводы:

При обучении моделей в целом лучше себя показал Градиентный бустинг, а модели в целом лучше обучались одинаково на датасете со всеми полями и датасете с отобранными полями.