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 выживаемости пассажиров при крушении “Титаника” с применением моделей машинного обучения

Студент ____ИУ5-34М____ _____ ___**Саврасов П.А.**___
(Группа) (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___

_____Саврасов Павел Анатольевич_____
(Фамилия, имя, отчество)

Тема курсового проекта: __Прогнозирование выживаемости пассажиров при крушении ____
“Титаника” 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:

Расчётно-пояснительная записка на _____ листах формата А4.
Перечень графического (иллюстративного) материала (чертежи, плакаты, слайды и т.п.)
_____
_____
_____
_____

Дата выдачи задания « __ » _февраля_ 2021 г.

Руководитель курсового проекта _____ ____Ю.Е. Гапанюк____
(Подпись, дата) (И.О.Фамилия)
Студент _____ __П.А. Саврасов_____
(Подпись, дата) (И.О.Фамилия)

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

Введение:

Крушение “Титаника” является одной из самых крупных техногенных катастроф, унёсшей жизни более чем 1.5 тысячи человек. Согласно данным, сохранившимся о крушении, при спасении потерпевших существовали некоторые критерии, по которым отбирались пассажиры.

Существует датасет, который содержит информацию о пассажирах “Титаника”. Ниже представлена его структура:

Имя поля	Значение поля	Расшифровка
survival	Выжил ли пассажир	0 = Нет 1 = Да
pclass	Класс	1 = первый класс 2 = второй класс 3 = третий класс
sex	Пол пассажира	
Age	Возраст пассажира	
sibsp	Число братьев или сестёр	
parch	Число детей или родителей	
ticket	Номер билета	
fare	Цена билета	
cabin	Номер каюты	
embarked	Порт отправления	C = Cherbourg Q = Queenstown S = Southampton

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')
```

```
data = pd.read_csv('titanic/train.csv', sep = ",")
data.head()
```

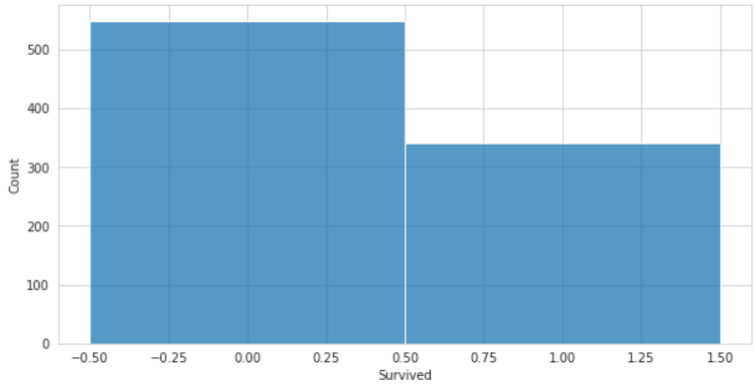
	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	NaN	S
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	C85	C
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250	NaN	S
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000	C123	S
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500	NaN	S

```
data.shape
```

```
(891, 12)
```

Размер датасета составляет 891 запись до обработки пустых полей. Рассмотрим количественное отношение целевого класса.

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

sns.histplot(data['Survived'], discrete=True)
```



Как видно из гистограммы, число выживших меньше, чем число погибших в представленном датасете.

Обработка пропущенных значений

```
print('Процент пустых данных по столбцам:')
data.isnull().sum()/data.shape[0]*100
```

```
Процент пустых данных по столбцам:

PassengerId     0.000000
Survived        0.000000
Pclass          0.000000
Name            0.000000
Sex             0.000000
Age            19.865320
SibSp           0.000000
Parch           0.000000
Ticket          0.000000
Fare            0.000000
Cabin          77.104377
Embarked        0.224467
dtype: float64
```

В датасете присутствуют пустые поля, которые необходимо обработать. В случае поля Каюта процент потерь составляет 77 процентов, поэтому просто исключим его из датасета. В случае поля возраст пустых значений 19 процентов. Заменить это поле средним значением не является решением, так как это сильно искажает распределение возрастов. Удалим записи с отсутствующим возрастом.
Поле Порт отправления с 0.2 процентами так же будет обработано удалением пустых значений.

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

data = data.dropna(subset=['Age'])

data = data.dropna(subset=['Embarked'])
data.head()
```

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Embarked
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	S
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	C
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250	S
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000	S
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500	S

```
data.shape
```

```
(712, 11)
```

Теперь в датасете 712 записей без пустых значений.

```
data.isnull().sum()
```

```
PassengerId    0
Survived       0
Pclass         0
Name           0
Sex            0
Age            0
SibSp          0
Parch          0
Ticket         0
Fare           0
Embarked       0
dtype: int64
```

Кодирование категориальных признаков

```
data.dtypes
```

```
PassengerId      int64
Survived         int64
Pclass           int64
Name            object
Sex             object
Age            float64
SibSp            int64
Parch            int64
Ticket          object
Fare           float64
Embarked        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(labelMap)
```

В датасете присутствуют не числовые значения. Введём функцию для преобразования их в числовой тип через кодирование индексами. В процессе работы функция выведет на экран связку индекса и значения.

```
fitTransrotmRepr(data, 'Sex')
fitTransrotmRepr(data, 'Embarked')
data.head()
```

```
{'male': 1, 'female': 0}
{'S': 2, 'C': 0, 'Q': 1}
```

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Embarked
0	1	0	3	Braund, Mr. Owen Harris	1	22.0	1	0	A/5 21171	7.2500	2
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	0	38.0	1	0	PC 17599	71.2833	0
2	3	1	3	Heikkinen, Miss. Laina	0	26.0	0	0	STON/O2. 3101282	7.9250	2
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	0	35.0	1	0	113803	53.1000	2
4	5	0	3	Allen, Mr. William Henry	1	35.0	0	0	373450	8.0500	2

```
data.dtypes
```

```
PassengerId      int64
Survived         int64
Pclass           int64
Name            object
Sex              int64
Age            float64
SibSp            int64
Parch            int64
Ticket          object
Fare           float64
Embarked         int64
dtype: object
```

Просмотр полученных данных

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

На диаграммах выше построены диаграммы распределений и зависимостей полей датасета.

Отбор признаков

Удаление ненужных признаков

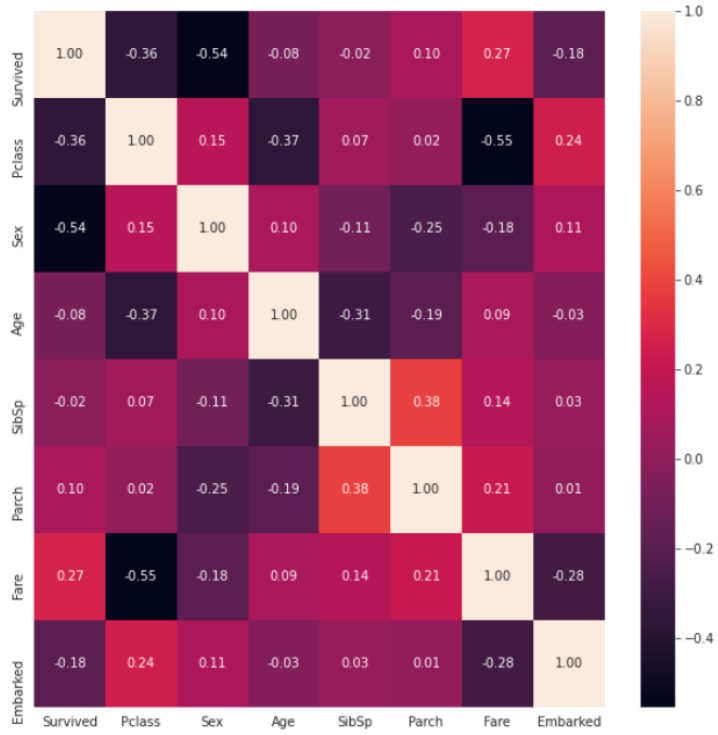
```
data.drop('Name', axis=1, inplace=True)
data.drop('Ticket', axis=1, inplace=True)
data.drop('PassengerId', axis=1, inplace=True)
```

Поля Имя, Идентификатор пассажира и Номер билета не несут полезной информации, поэтому удалим их из датасета.

Отбор на основе корреляции

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

```
<AxesSubplot:>
```



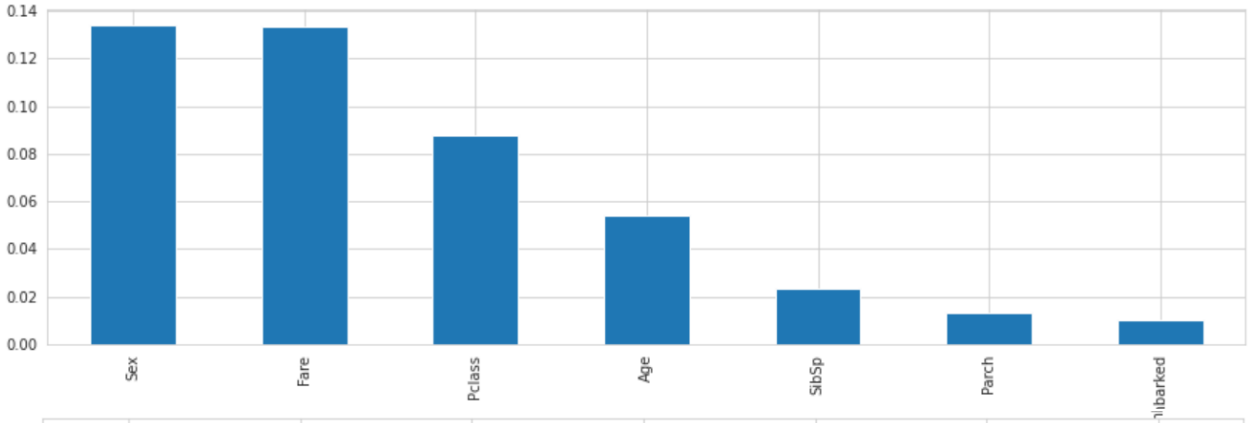
Из Корреляционной матрицы видно, что нет зависимостей между целевым полем и остальными полями. Присутствует лишь обратная зависимость между Уплаченой ценой и классом.

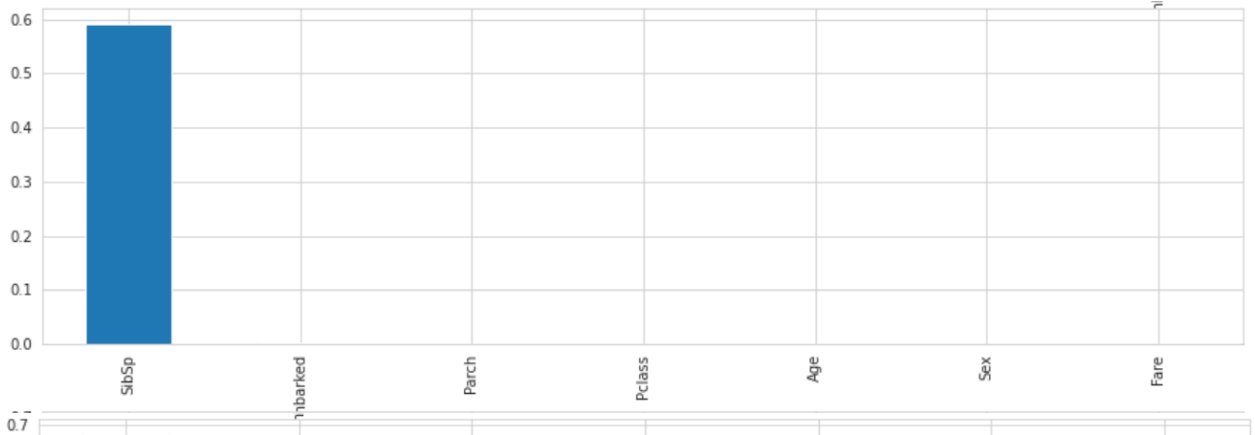
Отбор методом на основе статистических характеристик и методом вложений

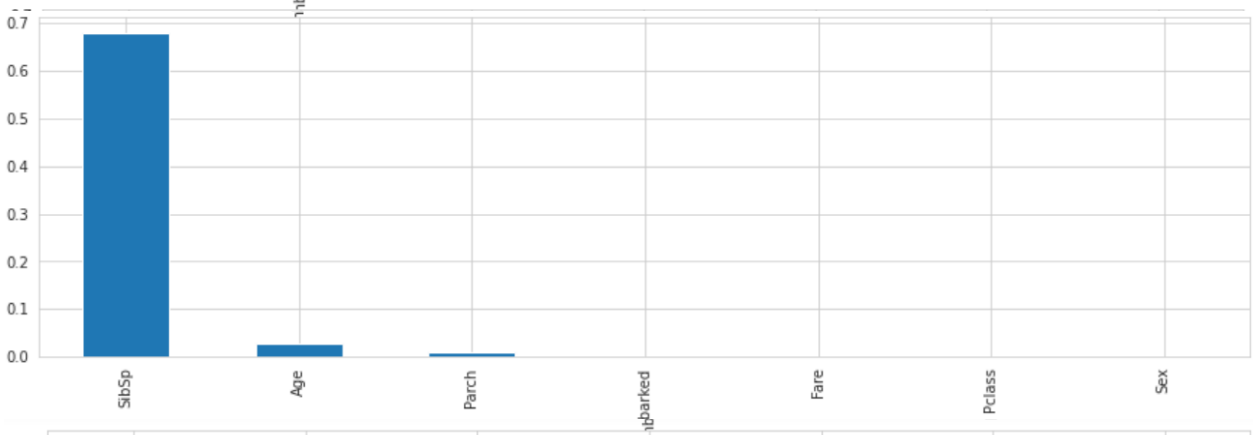
```
class FeatureSelector:
    def __init__(self, df, target, method, index, isModel):
        if isModel:
            model = method()
            model.fit(df.drop(columns=[target]), df[target])
            self.mic = model.feature_importances_

        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()
```

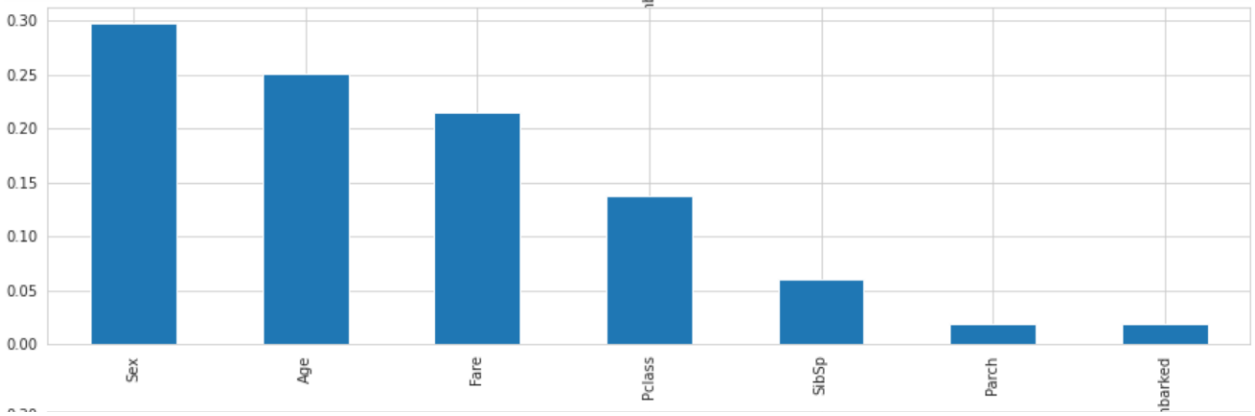
Введём класс для вывода статистики о важности полей с точки зрения различных методов их отбора.

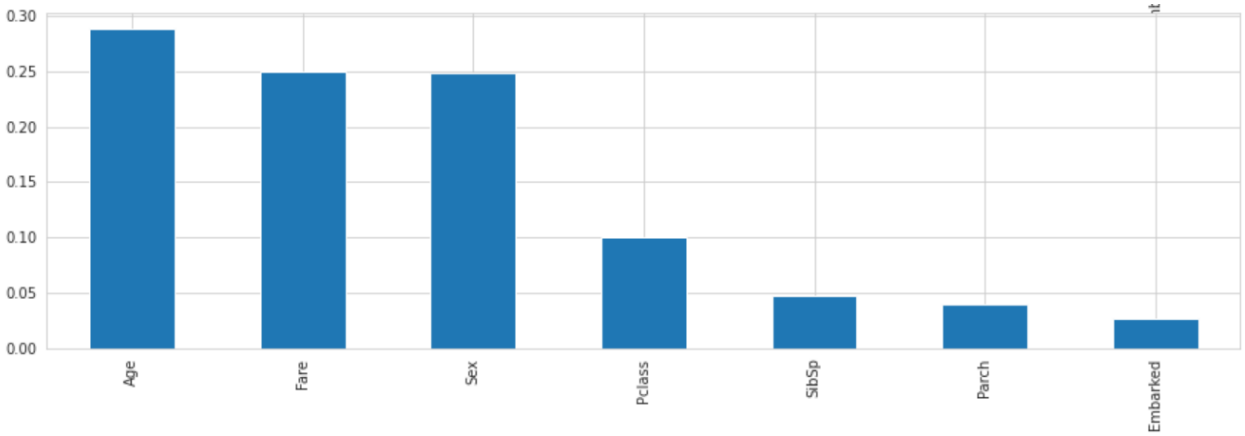
```
selectors = [
    FeatureSelector(data, 'Survived', mutual_info_classif, False, False),
    FeatureSelector(data, 'Survived', chi2, True, False),
    FeatureSelector(data, 'Survived', f_classif, True, False),
    FeatureSelector(data, 'Survived', DecisionTreeClassifier, False, True),
    FeatureSelector(data,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


0.14
0.12
0.10
0.08
0.06
0.04
0.02
0.00
Sex
Fare
Pclass
Age
SibSp
Parch


0.6
0.5
0.4
0.3
0.2
0.1
0.0
SibSp
Parch
Pclass
Age
Sex
Fare


0.7
0.6
0.5
0.4
0.3
0.2
0.1
0.0
SibSp
Age
Parch
Fare
Pclass
Sex

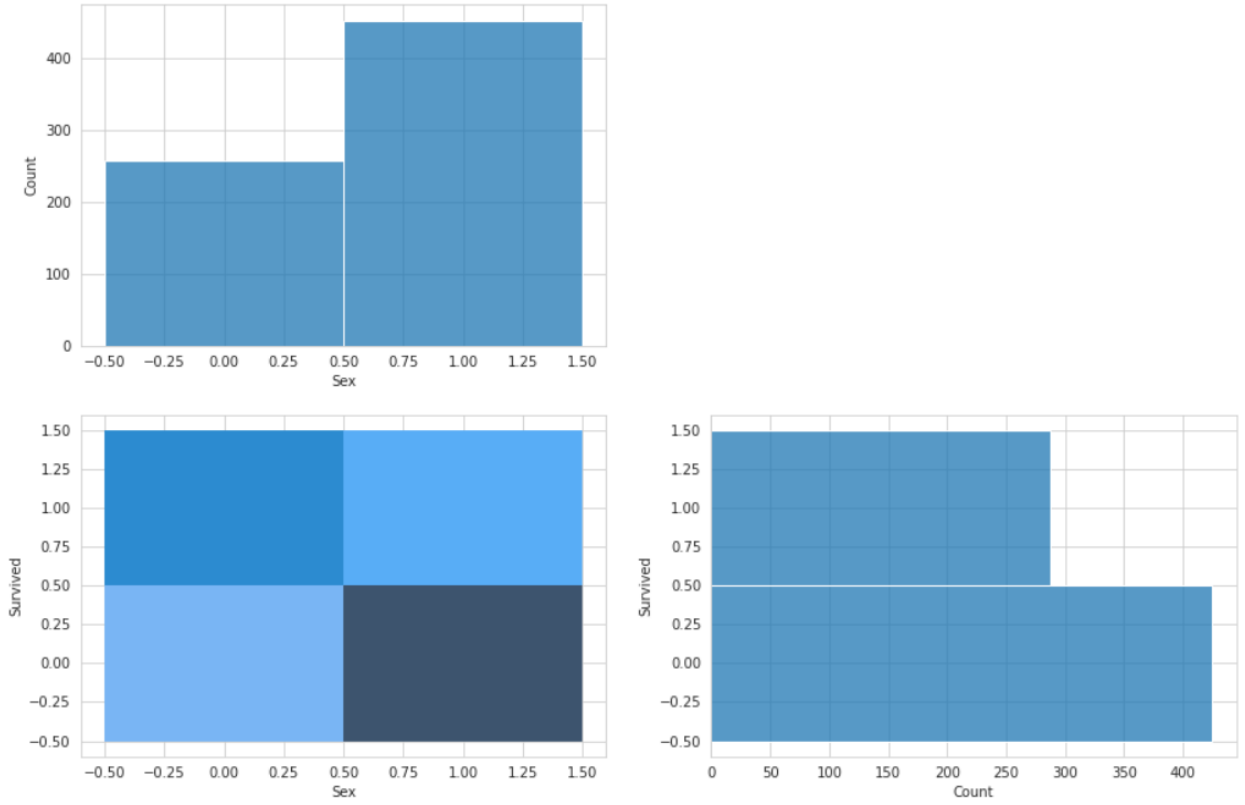

0.30
0.25
0.20
0.15
0.10
0.05
0.00
Sex
Age
Fare
Pclass
SibSp
Parch



Из полученных диаграмм видно, что важными полями оказались: Пол, Возраст, Уплаченная цена, Класс, и число детей. Рассмотрим как распределены эти поля по отношению к целевому признаку.

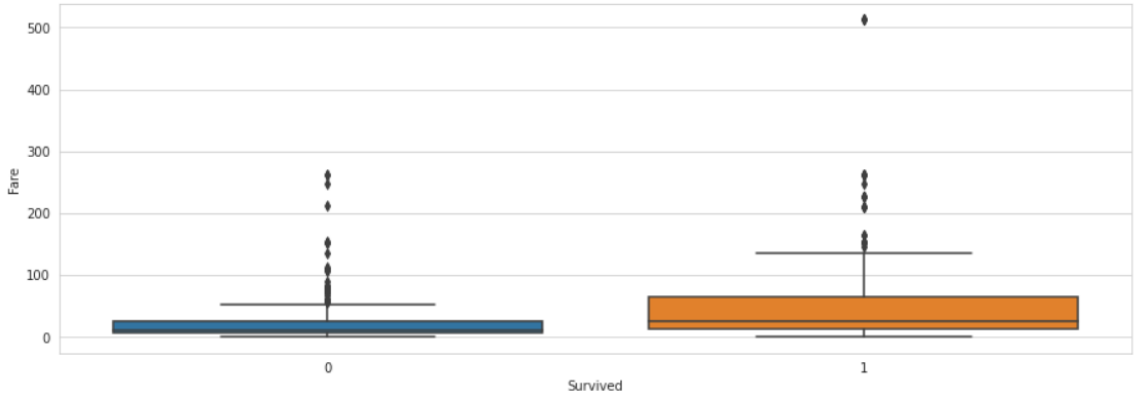
```
def joinhist(df, col, target):
    plt.subplots(figsize=(15,10))

    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, 'Sex', 'Survived')
```

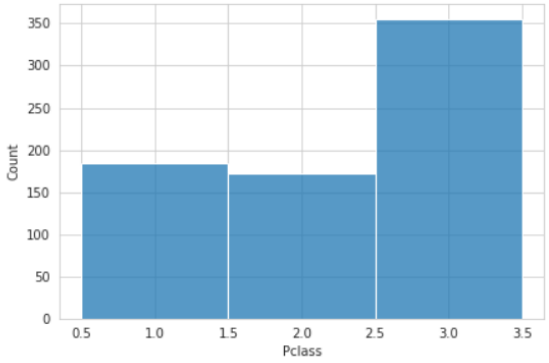


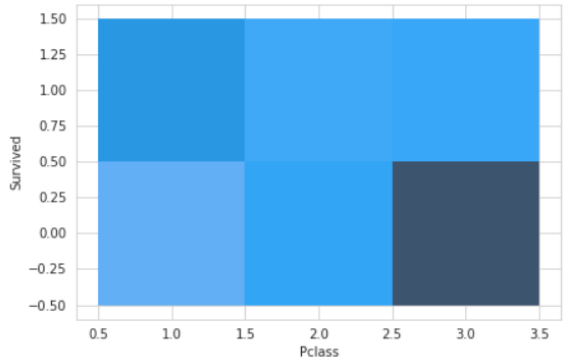
```
fig, ax = plt.subplots(figsize=(15,5))
sns.boxplot(x='Survived', y='Fare', data=data)
```

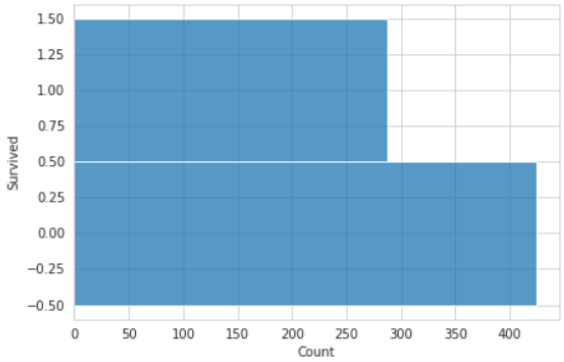
```
<AxesSubplot:xlabel='Survived', ylabel='Fare'>
```



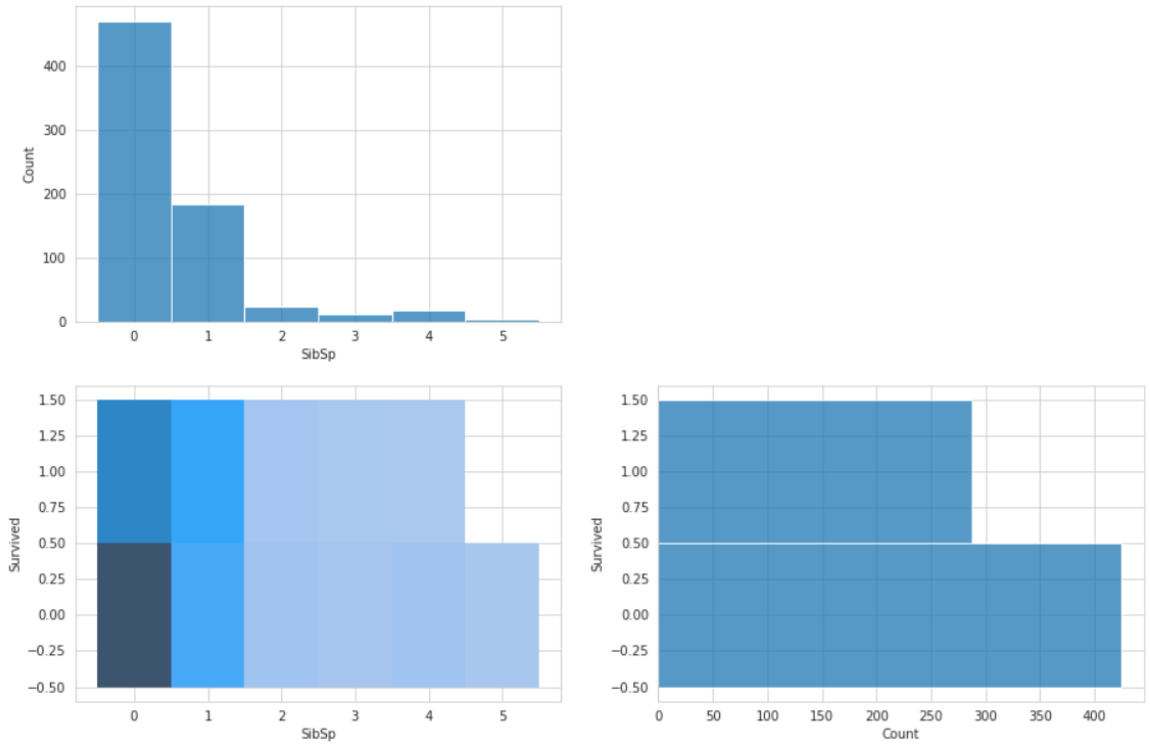

```
joinhist(data, 'Pclass', 'Survived')
```



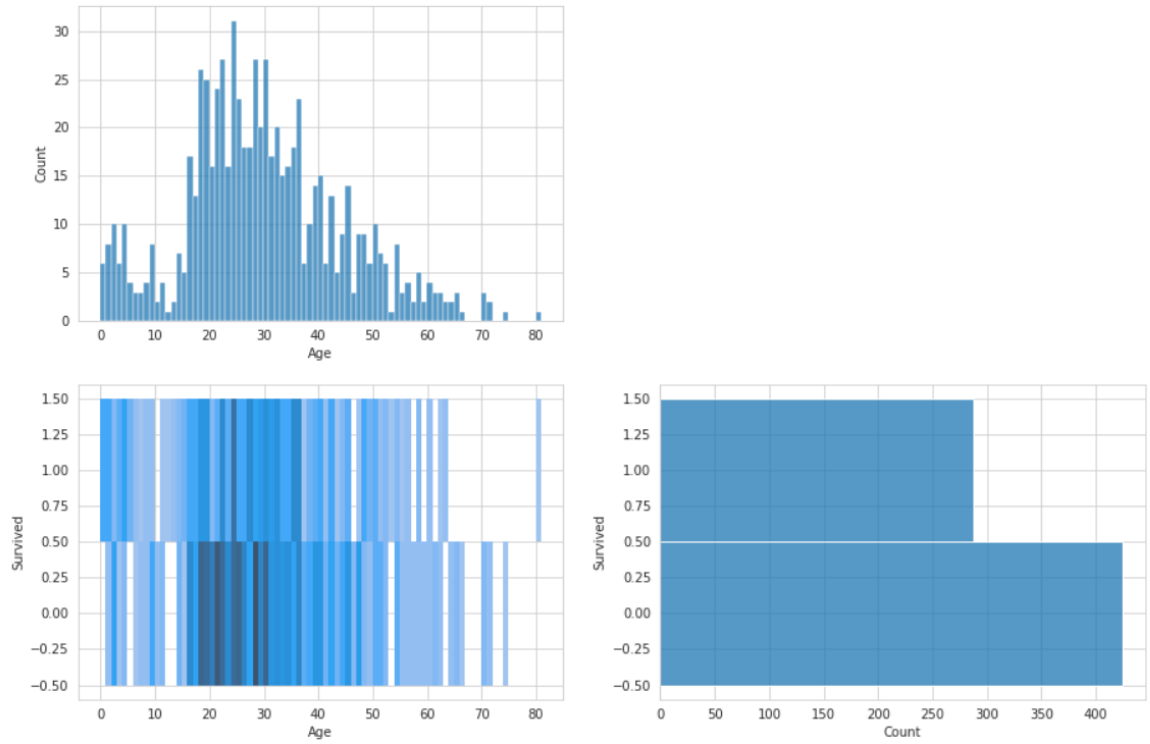




```
joinhist(data, 'SibSp', 'Survived')
```




```
joinhist(data, 'Age', 'Survived')
```




```
selectedFeatures = [
    'Sex',
    'Age',
    'Pclass',
    'SibSp'
]
```

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=['Survived']),
    data['Survived'], 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['Survived'],
    test_size=0.3,
    random_state = 1)
```

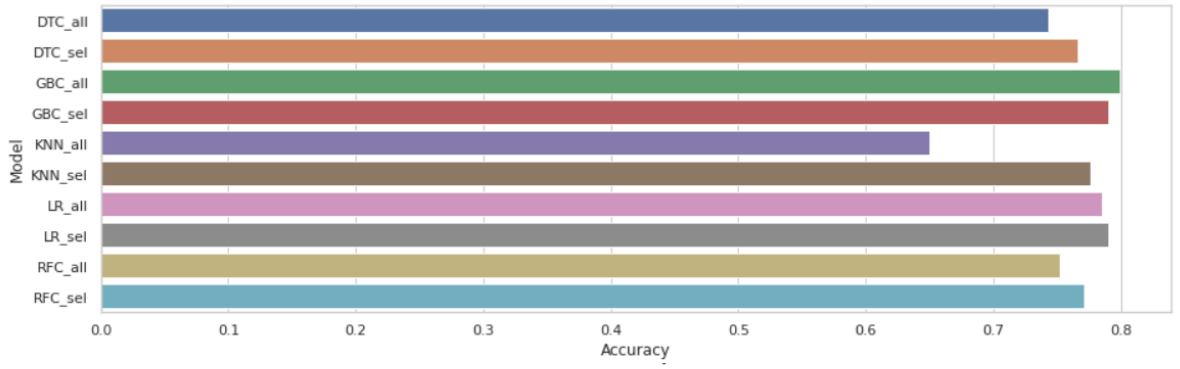
```
for modelName, model in models.items():
    tmp = [modelName + '_sel',]
    currModel = model()
    currModel.fit(xTrain, yTrain)
    yPred = currModel.predict(xTest)
    tmp.append(accuracy_score(yTest, yPred))
    tmp.append(precision_score(yTest, yPred))
    tmp.append(recall_score(yTest, yPred))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

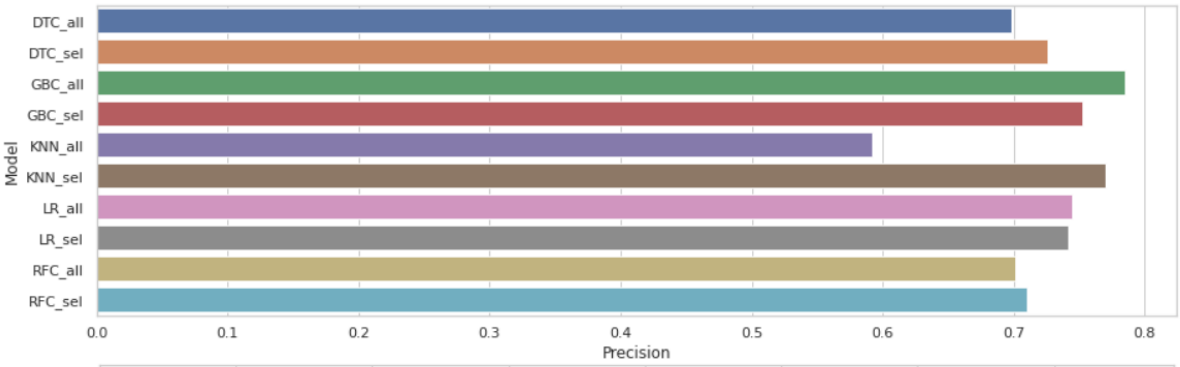
```
dfReport = pd.DataFrame(report, columns=['Model','Accuracy','Precision','Recall'])
dfReport = dfReport.sort_values(by=['Model'])
dfReport.head(10)
```

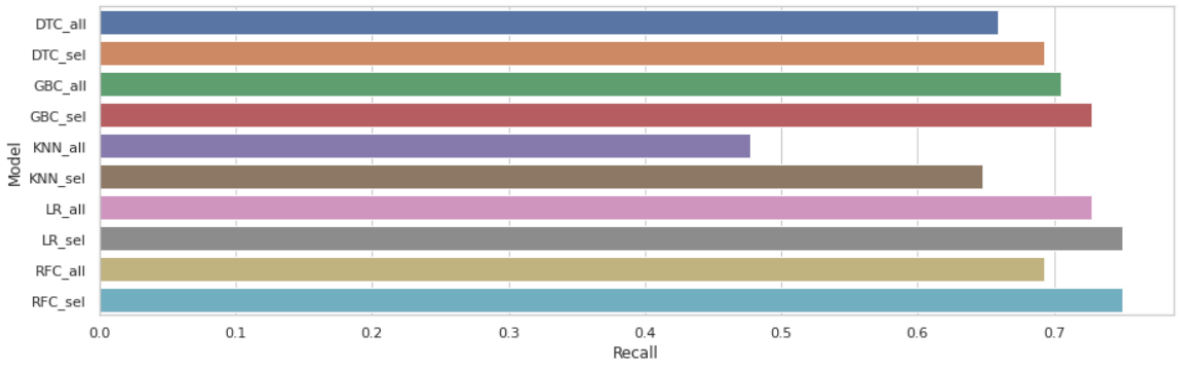
	Model	Accuracy	Precision	Recall
2	DTC_all	0.742991	0.694118	0.670455
7	DTC_sel	0.761682	0.717647	0.693182
4	GBC_all	0.799065	0.784810	0.704545
9	GBC_sel	0.789720	0.752941	0.727273
0	KNN_all	0.649533	0.591549	0.477273
5	KNN_sel	0.775701	0.770270	0.647727
1	LR_all	0.785047	0.744186	0.727273
6	LR_sel	0.789720	0.741573	0.750000
3	RFC_all	0.761682	0.712644	0.704545
8	RFC_sel	0.766355	0.697917	0.761364

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:

Результаты отбора параметров показали, что наиболее важным оказались Пол, Возраст, Класс, и Количество детей.

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