# **Задачи**

* Проведите базовый анализ данных, разделите данные на обучение/тест, разработайте baseline модель решающего дерева [DecisionTreeClassifier](https://scikit-learn.org/stable/modules/generated/sklearn.tree.DecisionTreeClassifier.html), оцените работу модели, отобразите важности признаков;
* Разработайте модель линейной регрессии, оцените и сравните с моделью дерева;
* Произведите стандартизацию численных признаков и оцените (сравните) работу моделей с результатами обучения без стандартизации;
* Изучите влияние max\_depth и criterion на показатели дерева, попробуйте 5 разных значений для каждого критерия, оцените с помощью кросс-валидации на обучающей выборке, сделайте таблицу;
* Проведите расширенный анализ данных, выберите наиболее приоритетные для классификации признаки, сравните выбранные признаки с показателями важности признаков, создайте новые признаки; В результате расширенного анализа обратите внимание на следующие особенностями:
  + Проведите создание новых признаков, добавляя каждый новый признак проведите оценку модели:
    - FamilySize - размер родственников на корабле;
    - IsAlone - является ли пассажир один на корабле или нет (бинарный признак);
    - FarePerPerson - оплата билета на человека в семье (воспользоваться FamilySize);
    - \*NameTitle (если придумаете как) - название титула, сформированое из признака Name, редкие титулы стоит объединить в одну группу;
  + Оцените работу модели при добавлении группировки признаков Age, Fare;
* Обучите модель дерева и модель леса [RandomForestClassifier](https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html) по подготовленным данным;
* Оцените влияние аргументов max\_depth и n\_estimators на точность модели (5 значений для каждого) с помощью кросс-валидации на обучающей выборке. Постройте таблицу зависимости метрик от величин;
* Определите наилучшие параметры для модели случайного леса через [GridSearchCV](https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.GridSearchCV.html);
* Определите наилучшие параметры для модели случайного леса через [RandomizedSearchCV](https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.RandomizedSearchCV.html);
* Постройте лучшие модели леса и решающего дерева и сравните их по показателям на выборке для теста.
* Примените подход [Recursive Feature Elimination](https://scikit-learn.org/stable/modules/generated/sklearn.feature_selection.RFE.html) на лучшую модель случайного леса. Сравните оценку важности признаков RFE и то, что показывает лес.
* Постройте ROC кривые моделей, сравните их и сделайте выводы:
  + лучшая модель на всех признаках;
  + лучшая модель только на топ-7 лучших признаках по RFE;
  + лучшая модель на топ-5 лучших признаках по RFE.

import numpy as np

import pandas as pd

import matplotlib.pyplot as plt

import seaborn as sns

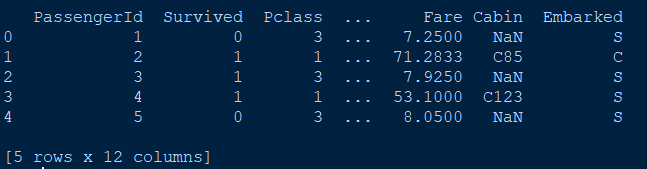
import warnings

warnings.filterwarnings('ignore')

data = pd.read\_csv('C:\ml\_edu-master\datasets\Titanic\_train.csv')

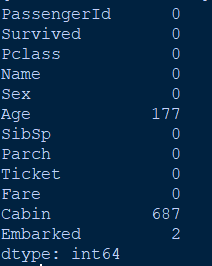
print(data.head())

Результат



print(data.isnull().sum())

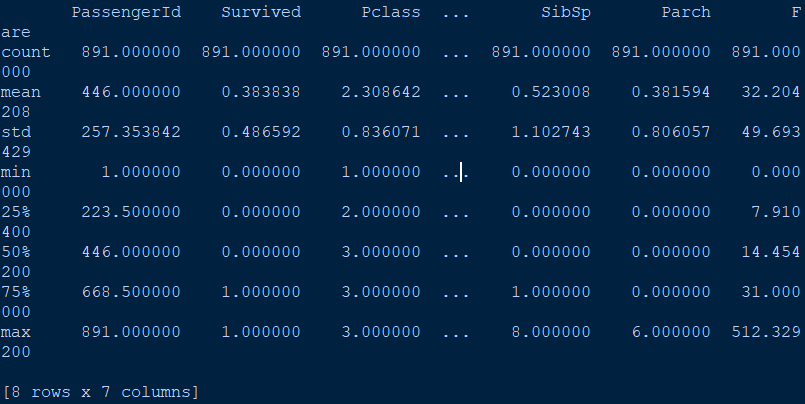
Результат



Первый взгляд на нет пропущенного значения, как видно из приведенного выше рисунка, возраст (Age), кабина(cabin)и встроенный(boarding port) есть пропущенное значение трех столбцов, в которых кабина(cabin) пропущенное значение относительно велико, потому что значение этого столбца само по себе не велико, мы откажемся позже.

print(data.describe())

Результат



Давайте рассмотрим конкретное распределение данных через данные.describe () может видеть, что данные составляют в общей сложности 891 строку, вы также можете видеть максимальные и минимальные значения некоторых столбцов, например возраст этого столбца, самый старый-80 лет, самый маленький-менее одного года.

fig,ax = plt.subplots(1,2,figsize=(18,8))

#data['Survived'].value\_counts().plot.pie(explode=[0,0.1],autopct='%1.1f%%',shadow=True,ax=ax[0])

ax[0].pie(data['Survived'].value\_counts(),explode=[0,0.1],autopct='%1.1f%%',shadow=True,labels=[0,1])

ax[0].set\_title('Survived')

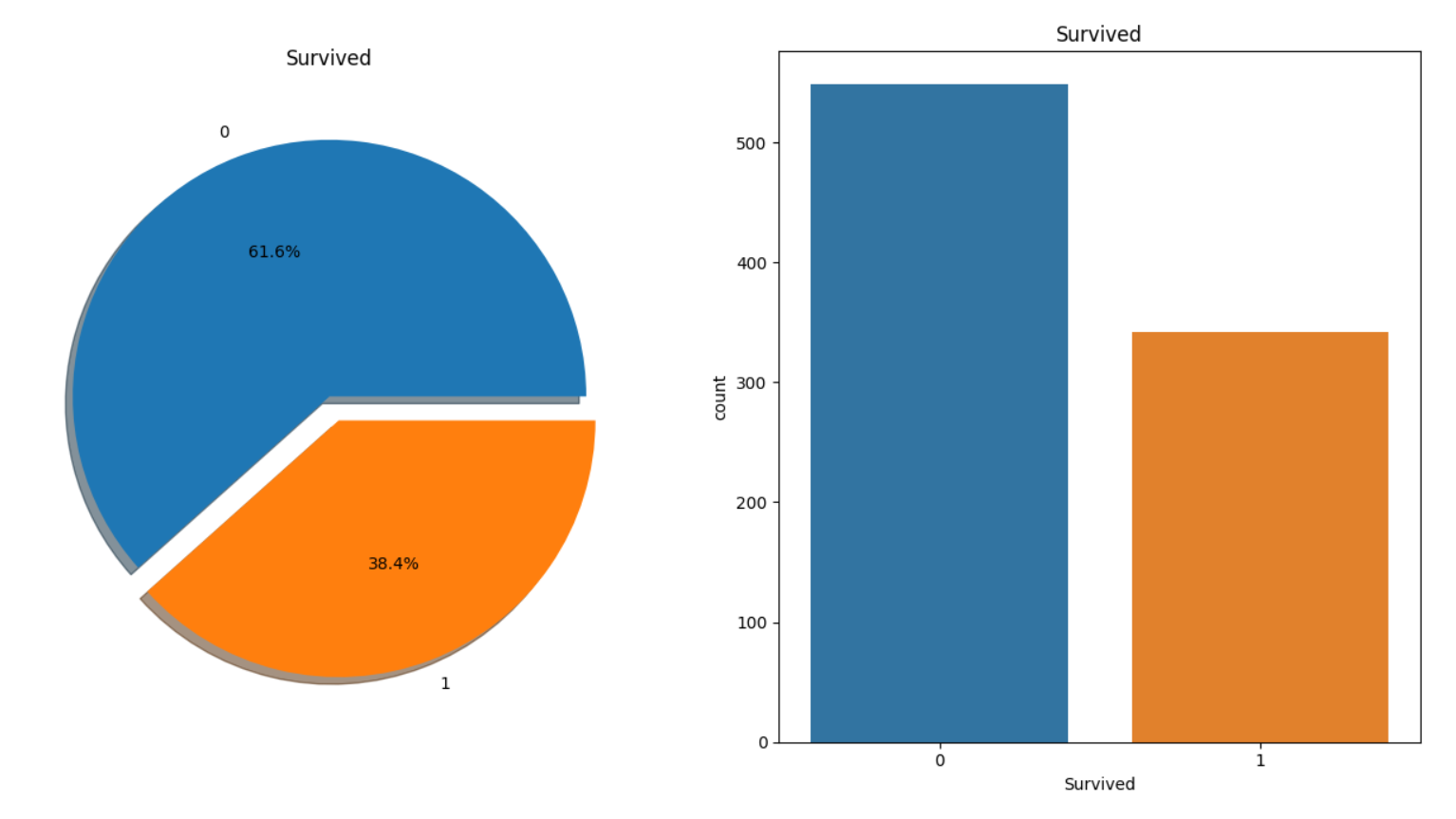
ax[0].set\_ylabel('')

sns.countplot('Survived',data=data,ax=ax[1])

ax[1].set\_title('Survived')

plt.show()

Результат

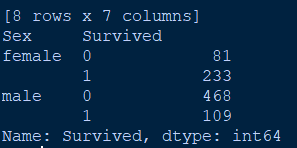


Через круговую диаграмму видно, что спасенные составили 38,4%, в общей сложности 891 человек, спасено более 300 человек, что свидетельствует о том, что показатель спасения относительно низок.

1. Анализ гендерных характеристик

print(data.groupby(['Sex','Survived'])['Survived'].count())

Результат



Согласно приведенному выше статистическому анализу можно видеть, что вероятность спасения женщин очень велика,мы также можем нарисовать картину, чтобы увидеть.

fig,ax = plt.subplots(1,2,figsize=(10,8))

data[['Sex','Survived']].groupby(['Sex']).mean().plot.bar(ax = ax[0])

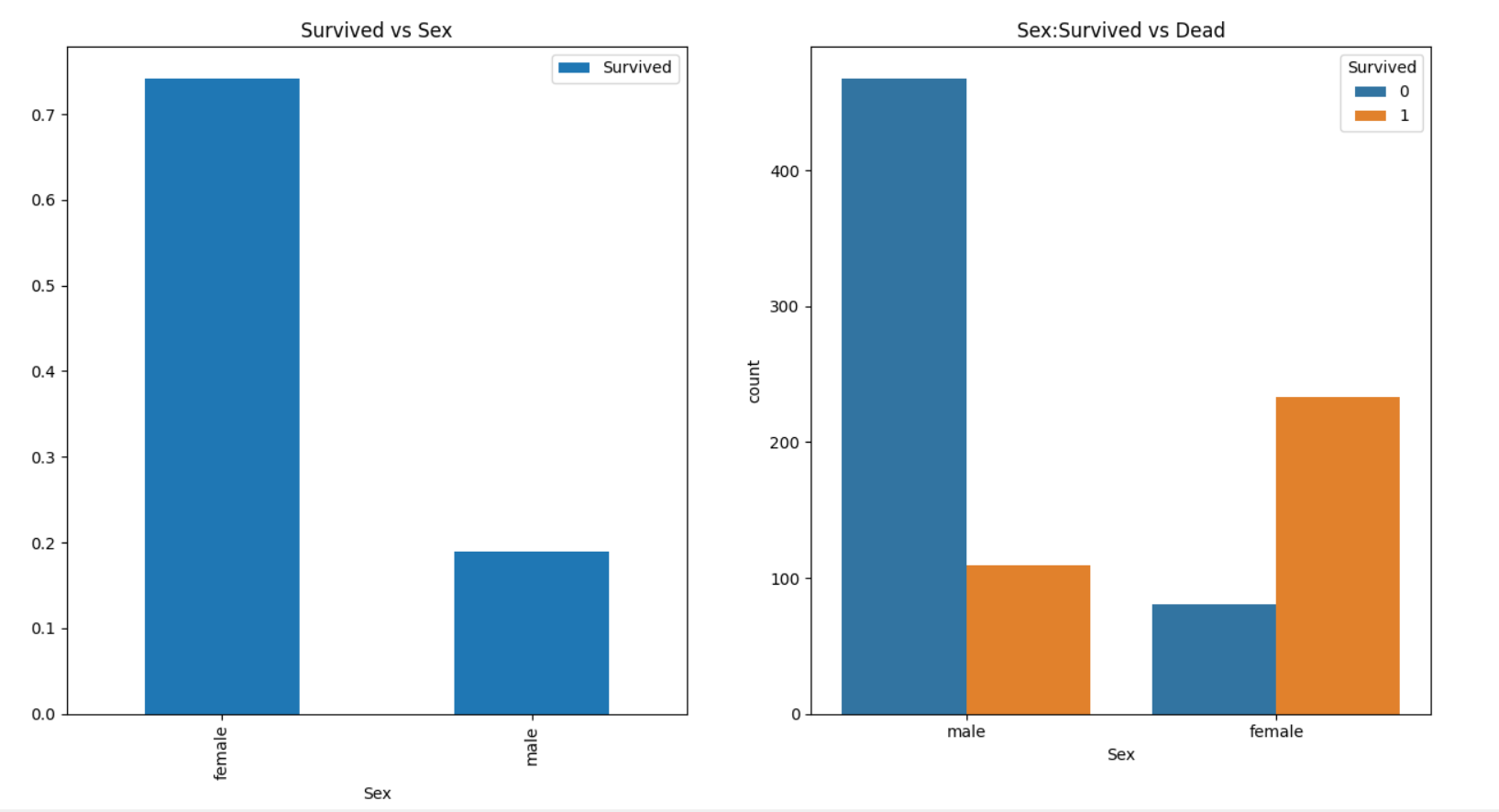
ax[0].set\_title('Survived vs Sex')

sns.countplot(x='Sex',hue='Survived',data=data,ax=ax[1])

ax[1].set\_title('Sex:Survived vs Dead')

plt.show()

Результат



Как видно из левой цифры, если это женщина, то есть более 70% шансов, что все женщины будут спасены, а мужчины только менее 20%, из правой цифры видно, что число спасенных женщин более чем в два раза больше, чем мужчин.В ходе анализа было установлено, что гендер является важным признаком.

1. Анализ характеристик класса кабины

fig,ax=plt.subplots(1,2,figsize=(10,8))

data['Pclass'].value\_counts().plot.bar(color=['green','blue','yellow'],ax=ax[0])

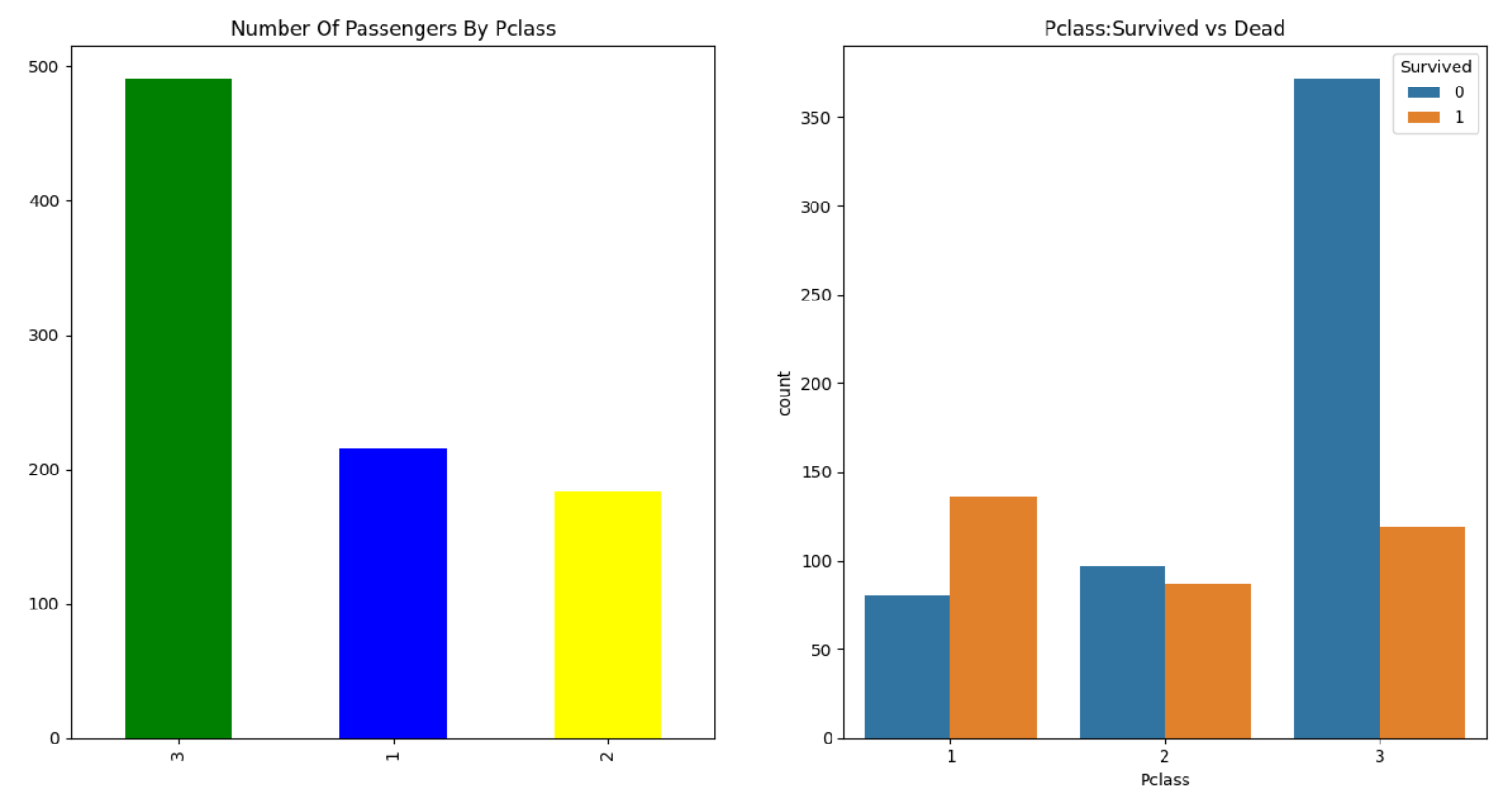
ax[0].set\_title('Number Of Passengers By Pclass')

sns.countplot('Pclass',hue='Survived',data=data,ax=ax[1])

ax[1].set\_title('Pclass:Survived vs Dead')

plt.show()

Результат



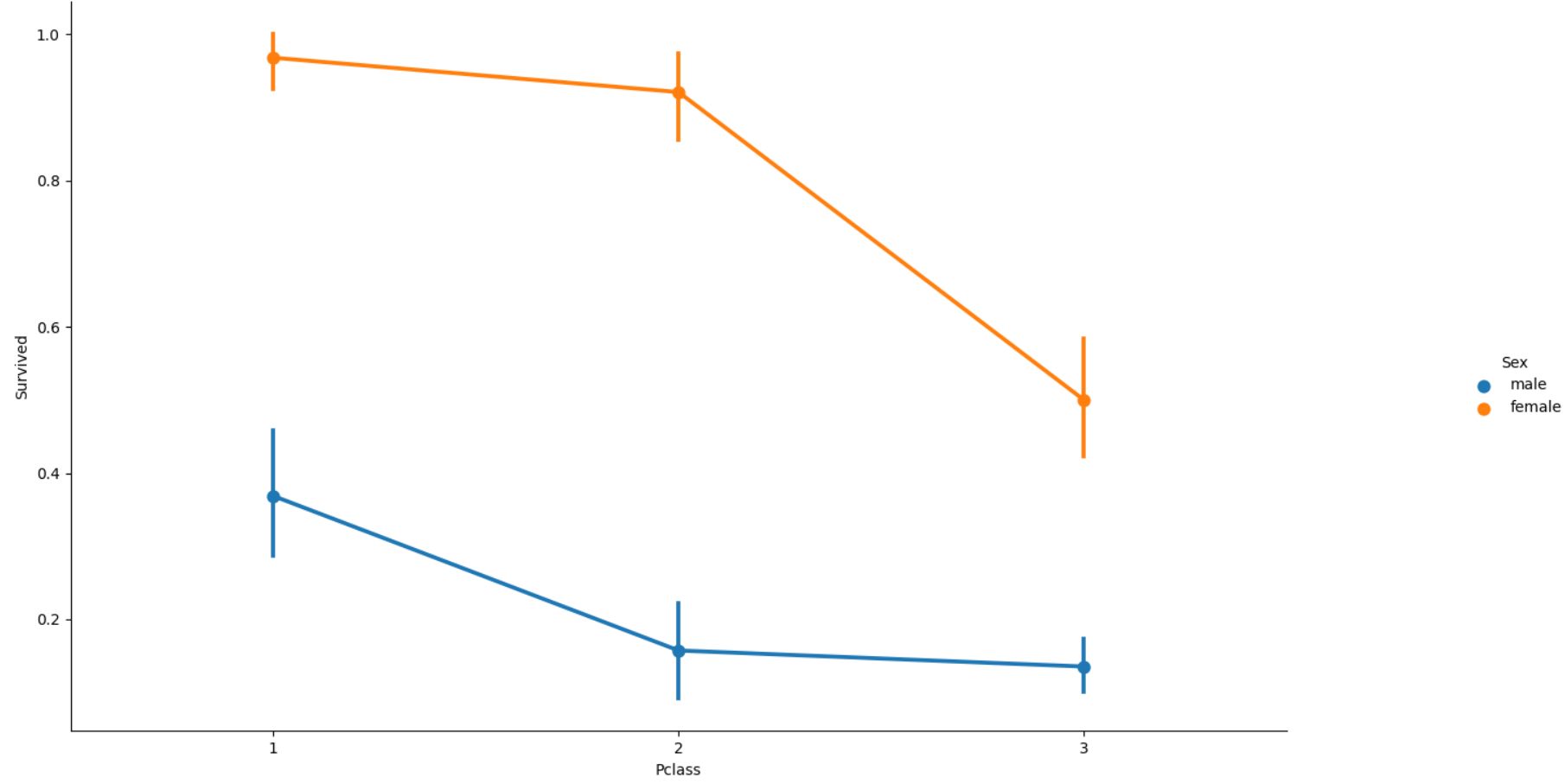
Количество людей, спасенных в салоне первого класса, больше, чем количество людей, которые не были спасены.Каюта второго класса примерно такая же.Количество людей, спасенных в салоне третьего класса, намного меньше, чем количество людей, которые не были спасены.У богатых по-прежнему гораздо больше шансов спастись, чем у бедных.

Давайте посмотрим на связь между полом и рангом каюты и спасением.

sns.factorplot('Pclass','Survived',hue='Sex',data=data)

plt.show()

Результат



Как видно из рисунка выше, доля женщин, спасенных в трех каютах, значительно выше, чем мужчин.

1. Анализ возрастных характеристик

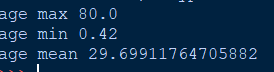
Давайте посмотрим на числовые характеристики возраста.

print('age max',data['Age'].max())

print('age min',data['Age'].min())

print('age mean',data['Age'].mean())

Результат



Затем нарисуйте и проанализируйте связь между классом кабины и возрастом, полом и возрастом, который нужно сохранить，

fig,ax = plt.subplots(1,2,figsize=(10,8))

sns.violinplot('Pclass','Age',hue='Survived',data=data,split=True,ax=ax[0])

ax[0].set\_title('Pclass and Age vs Survived')

ax[0].set\_yticks(range(0,110,10))

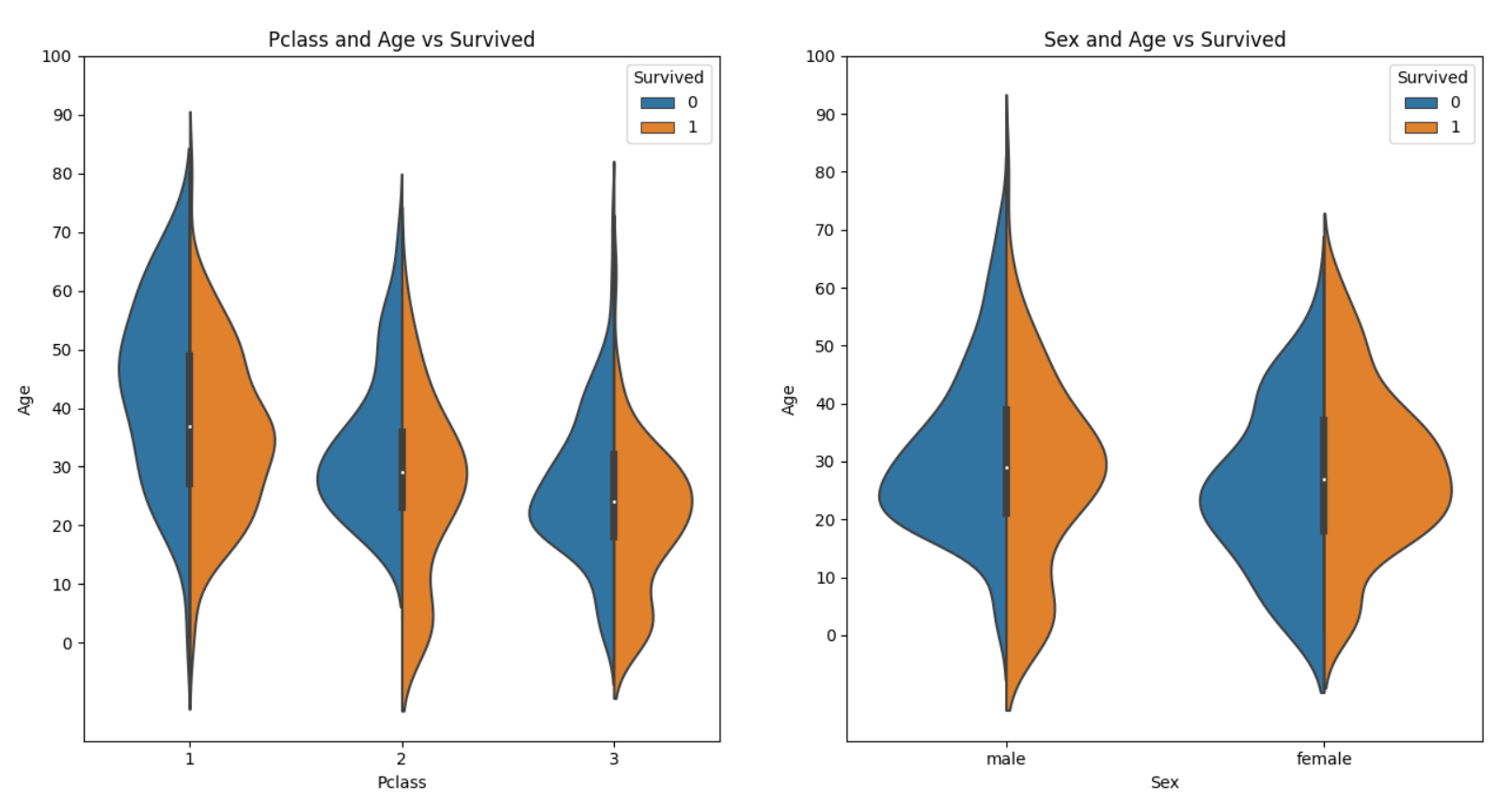
sns.violinplot('Sex','Age',hue='Survived',data=data,split=True,ax=ax[1])

ax[1].set\_title('Sex and Age vs Survived')

ax[1].set\_yticks(range(0,110,10))

plt.show()

Результат



Слева вы можете проанализировать, что кабина первого класса была спасена большей частью людей, сосредоточенных примерно в 37,8 лет, кабина второго класса была спасена большей частью людей, сосредоточенных в 30 лет, кабина третьего класса была сосредоточена примерно в 20-30 лет, вы можете видеть, что кабина первого класса была спасена старше, чем две другие кабины.Как видно из правильного рисунка, возраст самца был сохранен примерно в 30 лет, самка была сохранена примерно в 20 ~ 30 лет, разрыв не очень большой.

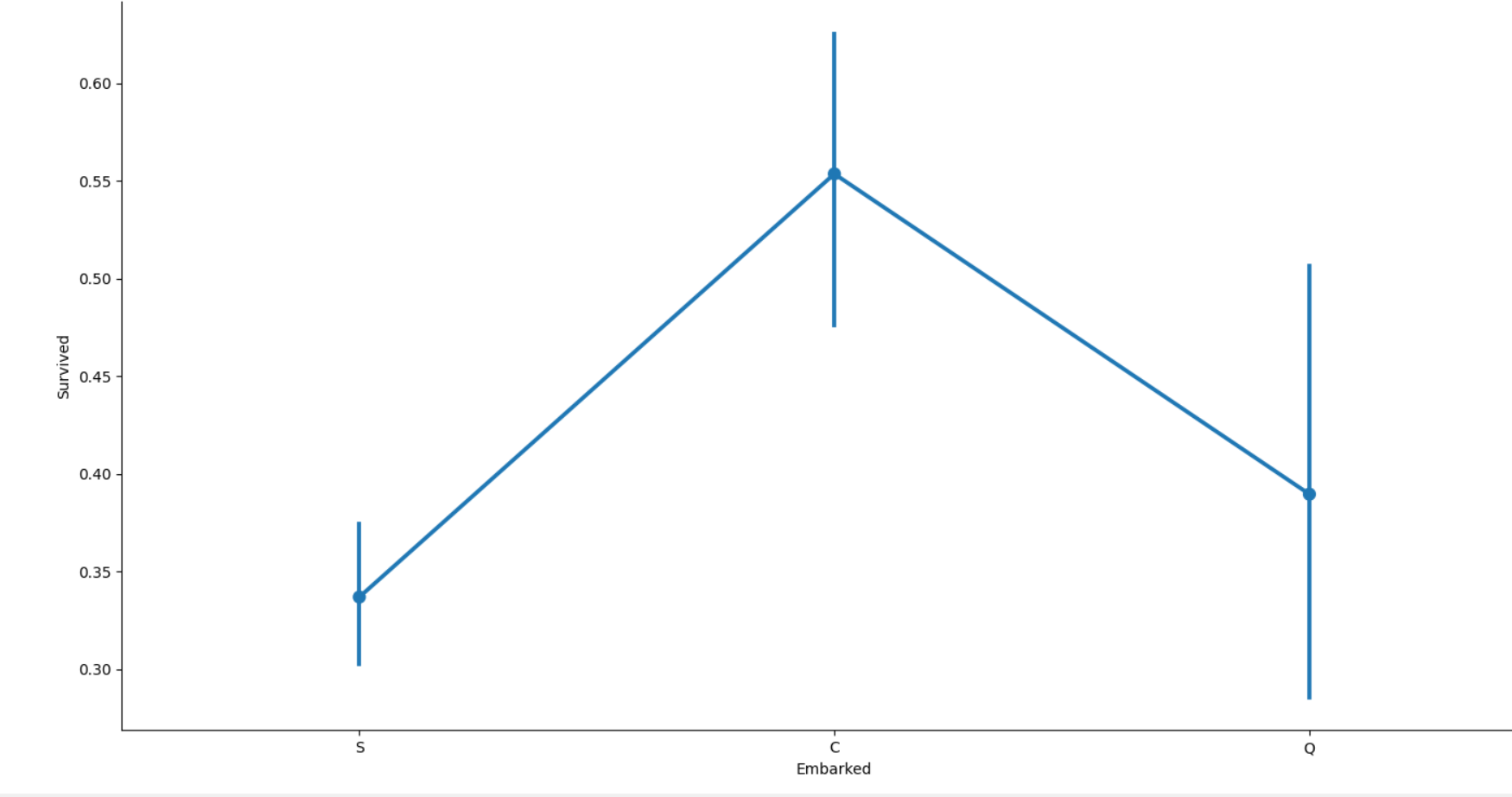
1. Aнализ особенностей места посадки на борт

sns.factorplot('Embarked','Survived',data=data)

fig=plt.gcf()

fig.set\_size\_inches(5,3)

Результат



Как видно из приведенного выше рисунка,выживаемость порта С является самой высокой, а Порта S-самой низкой

ig,ax=plt.subplots(2,2,figsize=(15,15))

sns.countplot('Embarked',data=data,ax=ax[0][0])

ax[0][0].set\_title('No. Of Passengers Boarded')

sns.countplot('Embarked',hue='Sex',data=data,ax=ax[0][1])

ax[0][1].set\_title('Male-Femal Split for Embarked')

sns.countplot('Embarked',hue='Survived',data=data,ax=ax[1][0])

ax[1][0].set\_title('Embarked vs Survived')

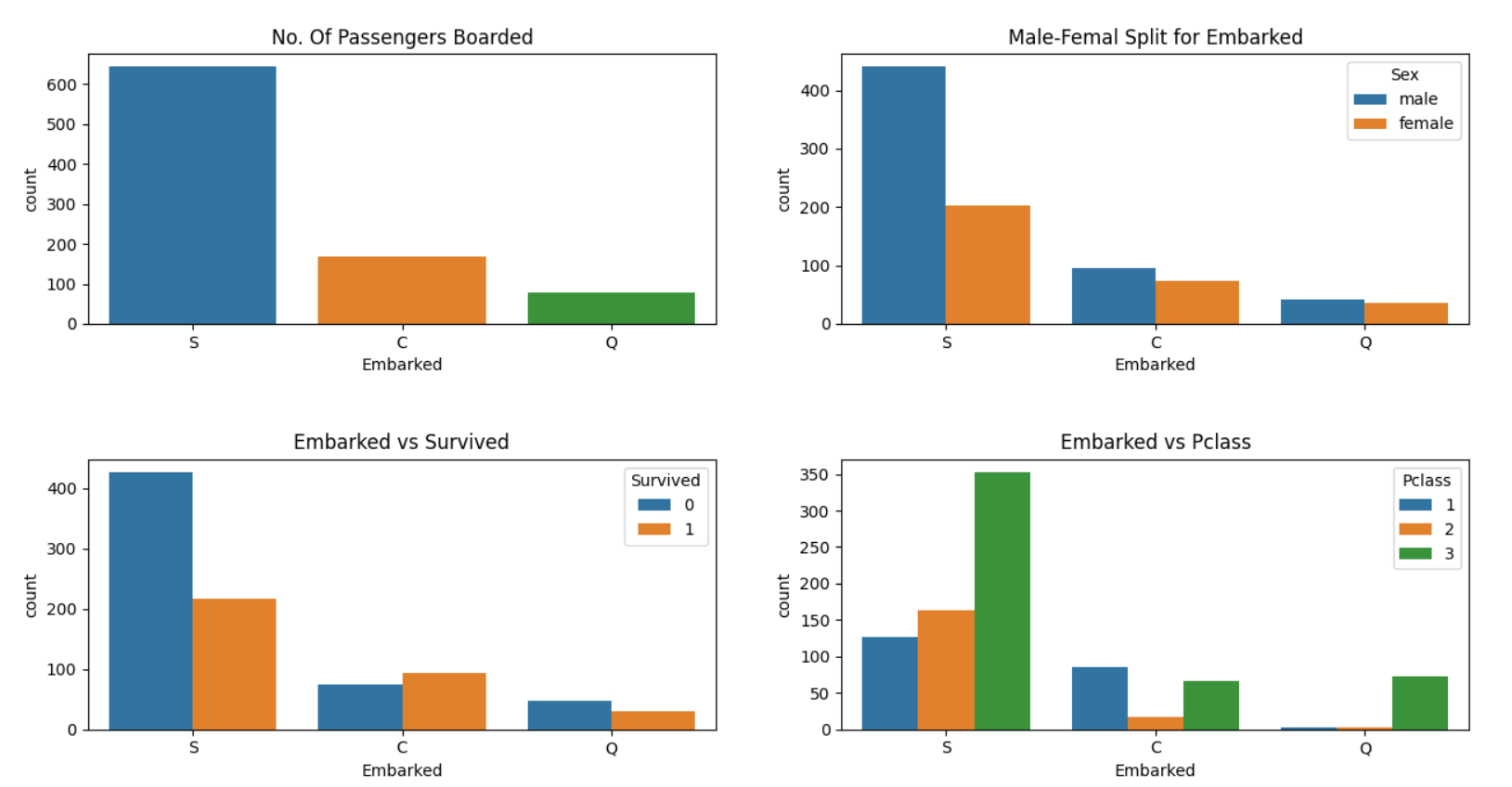
sns.countplot('Embarked',hue='Pclass',data=data,ax=ax[1][1])

ax[1][1].set\_title('Embarked vs Pclass')

plt.subplots\_adjust(wspace=0.2,hspace=0.5)

plt.show()

Результат

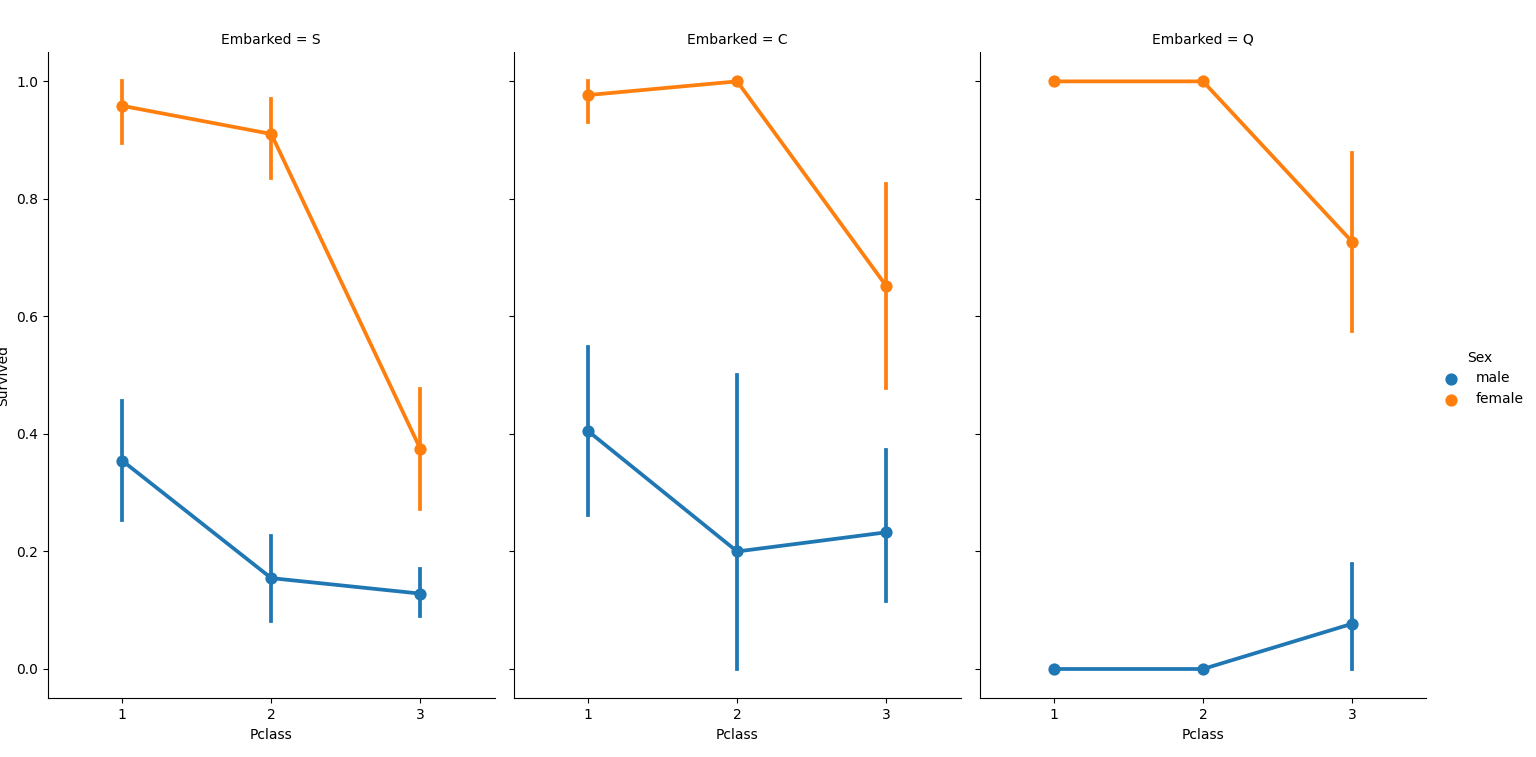


Вверху слева статистика количества людей на месте посадки, очевидно, что порт S является самым большим; вверху справа статистика количества мужчин и женщин на каждом месте посадки; внизу слева статистика количества спасенных и неуспешных людей на каждом месте посадки, в сочетании с приведенной выше таблицей, спасенных в порту S. Количество людей самое высокое, но процент спасенных - самый низкий. В правом нижнем углу статистика уровня кают для каждого места посадки показывает, что количество людей, садящихся на борт в порту S, самое большое, а количество людей в третьем классе - самое большое. Почти никто из пассажиров салона первого класса не садился в порт Q.

sns.factorplot('Pclass','Survived',hue='Sex',col='Embarked',data=data)

plt.show()

Результат



На приведенном выше рисунке показано соотношение мужчин и женщин, спасенных в каждой точке посадки и на каждом уровне каюты. Короче говоря, это полностью соответствует политике, касающейся женщин и детей.

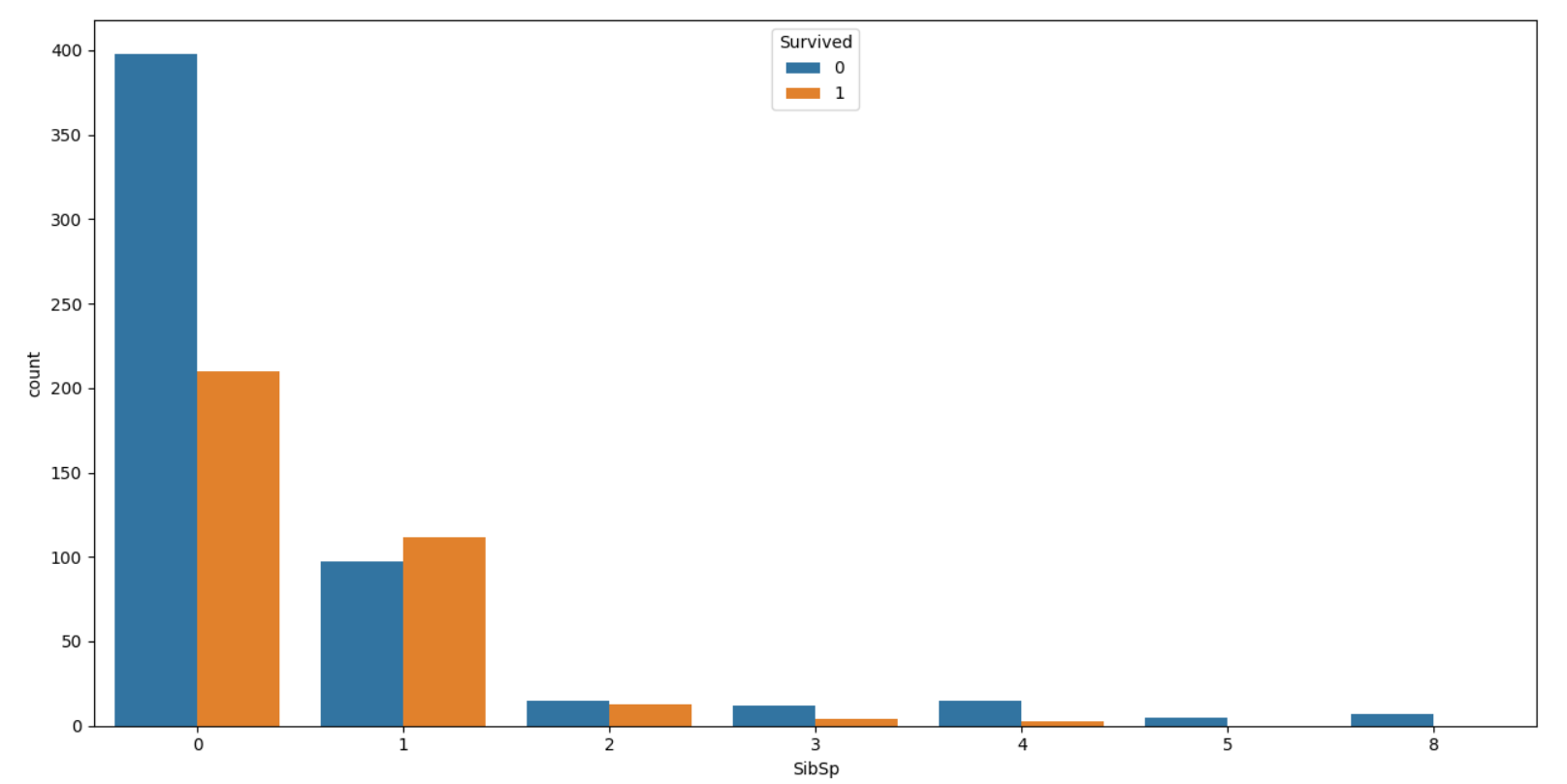
1. Анализ характеристик двоюродного брата / сестры

plt.figure(figsize=(8,8))

sns.countplot('SibSp',hue='Survived',data=data)

plt.show()

Результат



Из таблицы выше мы видим, что чем больше кузенов / сестер, тем меньше их число и меньше спасенных.

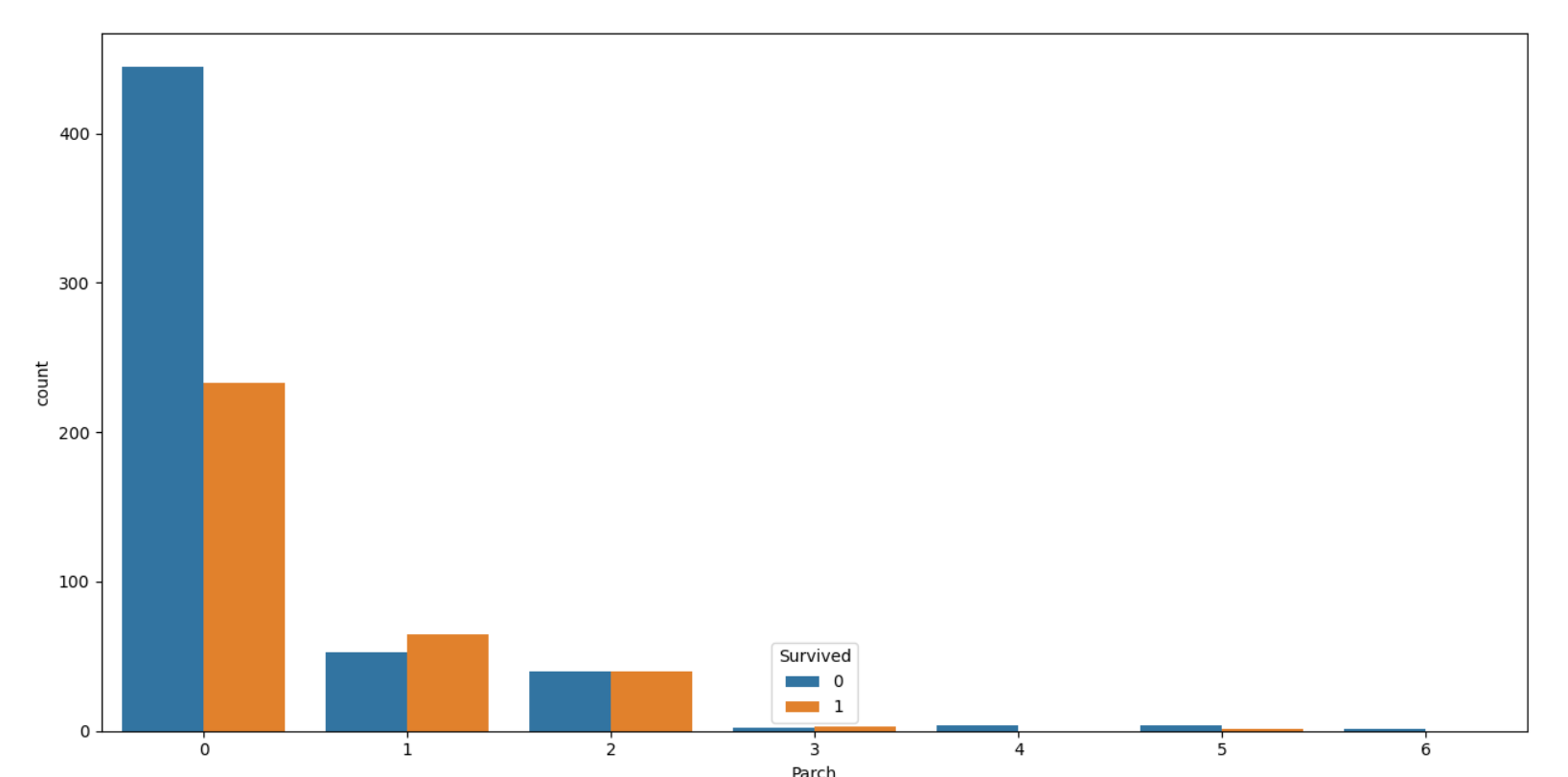
1. Количество родителей и детей характерный анализ

plt.figure(figsize=(8,8))

sns.countplot('Parch',hue='Survived',data=data)

plt.show()

Результат



Эта функция аналогична функции количества двоюродных братьев и сестер. Чем больше родителей и детей, тем меньше базовое число и меньше количество спасенных. Например, очень мало людей с 4 или 5 родителями или детьми, и они не спасены.

**Очистка и предварительная обработка данных**

1. Заполнение недостающего значения

Прежде всего, давайте разберемся с недостающими значениями, используя приведенные выше data.isnull().sum() мы знаем, что отсутствуют два собственных значения (Cabin не рассматривается).

Возрастные проблемы

О заполнении возраста, мы обычно выбираем средний возраст для заполнения, но здесь мы обнаружили, что в названии этой колонки name относительно длинное, середина будет с Mr,Miss,Mrs другими характеристиками, чтобы идентифицировать всех, мы можем в соответствии с этой функцией Найти каждый логотип, средний возраст для заполнения, так что эффект заполнения будет более практичным.

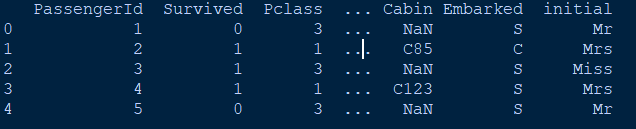
data['initial'] = 0

for i in data:

data['initial']=data.Name.str.extract('([A-Za-z]+)\.')

print(data.head())

Результат

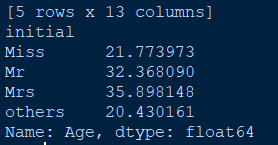


Теперь давайте посмотрим на количество мужчин и женщин.

data['initial']=data['initial'].replace(['Capt','Col','Countess','Don','Dr','Jonkheer','Lady','Major','Master','Mlle','Mme','Ms','Rev','Sir'],'others')

print(data.groupby('initial')['Age'].mean())

Результат



Наконец-то мы можем его заполнить.

data.loc[(data.Age.isnull())&(data.initial == 'Miss'),'Age']=22

data.loc[(data.Age.isnull())&(data.initial == 'Mr'),'Age']=32

data.loc[(data.Age.isnull())&(data.initial == 'Mrs'),'Age']=36

data.loc[(data.Age.isnull())&(data.initial == 'others'),'Age']=20

fig,ax = plt.subplots(1,2,figsize=(10,8))

data[data['Survived']==0].Age.plot.hist(ax=ax[0],bins=20,edgecolor='black',color='red')

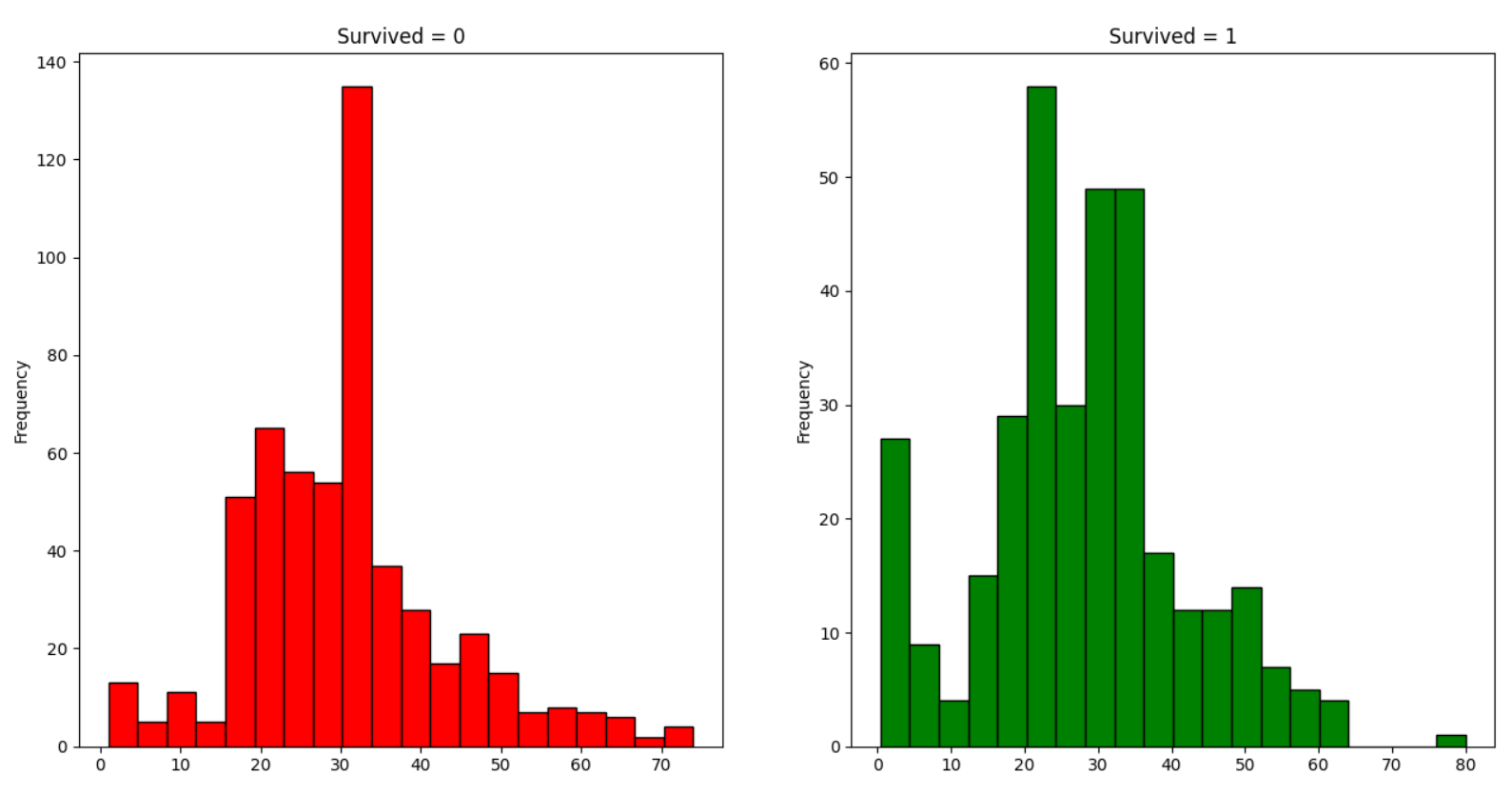
ax[0].set\_title('Survived = 0')

data[data['Survived']==1].Age.plot.hist(ax=ax[1],bins=20,edgecolor='black',color='green')

ax[1].set\_title('Survived = 1')

plt.show()

Результат



Как видно из приведенного выше рисунка, возрастное распределение большинства спасенных людей между 20~40, более 60 лет, чтобы спастись, относительно невелико(самый старый человек (80 лет)был спасен).

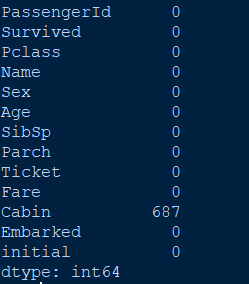
Заполнение отсутствующего значения **Embarked**

**Embarked заполнение отсутствующего значения столбца относительно просто,с помощью приведенного выше анализа функций мы можем обнаружить,что наибольшее количество портов S, поэтому мы используем большинство заполнений，**

**data['Embarked'].fillna('S',inplace=True)**

**print(data.isnull().sum())**

Результат



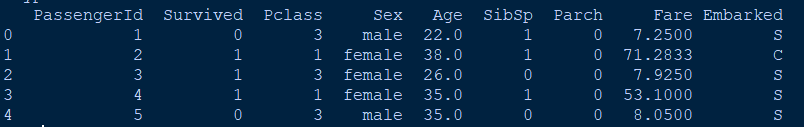
1. Предварительная обработка данных

Во-первых, давайте разберемся с бесполезными функциями.

data=data.drop(['Name','Ticket','Cabin','initial'],axis=1)

print(data.head())

Результат



#male=0,female=1

data.loc[data['Sex']=='male','Sex']=0

data.loc[data['Sex']=='female','Sex']=1

#S=0,C=1,Q=2

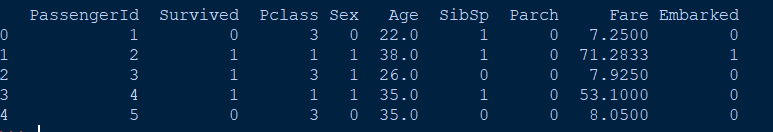
data.loc[data['Embarked']=='S','Embarked']=0

data.loc[data['Embarked']=='C','Embarked']=1

data.loc[data['Embarked']=='Q','Embarked']=2

data.head()

Результат



**Построение модели**

1. прогнозирование линейной регрессии

from sklearn.linear\_model import LinearRegression

from sklearn.model\_selection import KFold

linear = LinearRegression()

features = ['Pclass','Sex','Age','SibSp','Parch','Fare','Embarked']

kf=KFold(n\_splits=3,shuffle=False,random\_state=1)

source\_x = data[features]

source\_y = data['Survived']

print(source\_x.shape)

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

train\_x,test\_x,train\_y,test\_y = train\_test\_split(source\_x,source\_y,train\_size=0.8)

print('A：',source\_x.shape,'B：',train\_x.shape,'C',test\_x.shape)

Результат



linear.fit(train\_x,train\_y)

print(linear.score(test\_x,test\_y))



import numpy as np

scores = np.concatenate(scores,axis=0)

scores[scores>.5]=1

scores[scores<.5]=0

accuracy = sum(scores==source\_y)/len(source\_y)

print(accuracy)



# **Вопросы**

* Какой метод оценки модели лучше использовать в данной работе? Разделение на обучение/тест или кросс-валидация? Можно/нужно ли применять их вместе?
* За что отвечают параметры max\_depth и n\_estimators в модели случайных лесов? Как они влияют на работу модели?
* В чем отличие GridSearch от RandomSearch?
* Как влияет стандартизация признаков на работу модели леса?
* Что такое "важность признака"? Есть ли аналоги показателя в моделях линейной и логистической регрессии?

***1.Какой метод оценки модели лучше использовать в данной работе? Разделение на обучение/тест или кросс-валидация? Можно/нужно ли применять их вместе?***

***2.За что отвечают параметры max\_depth и n\_estimators в модели случайных лесов? Как они влияют на работу модели?***

Случайный лес Случайный лес - это, по сути, набор множества деревьев решений, которые переобучены разными способами. Мы можем взять среднее значение результатов этих разных деревьев, чтобы уменьшить переобучение, что может уменьшить переобучение и сохранить дерево Умение предсказывать. Случайный лес может использоваться для регрессии или классификации.Он может быть вызван через модуль RandomForestRegressor (регрессия) или модуль RandomForestClassifier (классификация) sklearn.ensemble.

Шаги по построению случайного леса:

①Определите количество деревьев, используемых для строительства

②Самостоятельная выборка данных

③Создать дерево решений на основе нового набора данных

Чтобы построить случайную модель леса, первым делом необходимо определить количество деревьев в лесу и скорректировать его с помощью модели. Чем больше n\_estimators, тем лучше, но занимаемая память и время обучения и прогнозирования также соответственно увеличиваются, а предельное преимущество уменьшается, поэтому выбирайте максимально возможные n\_estimators в приемлемой памяти / времени. В sklearn значение n\_estimators по умолчанию равно 10.

Случайный лес называется случайным лесом, потому что он добавляет случайности при построении. ② и ​​③ - проявления случайности. Второй шаг - это самовыборка данных, то есть случайная выборка из n\_sample точек данных с заменой и отрисовка всего n\_sample раз. Емкость нового набора данных равна исходному набору данных, но извлеченные образцы часто отличаются от исходного набора данных. Обратите внимание, что набор данных, используемый в построенном дереве n\_estimators, является независимым самовыборочным, чтобы гарантировать, что все деревья отличаются друг от друга.

Следующий третий шаг - построить дерево решений на основе этого нового набора данных. Из-за добавления случайности конструкция отличается от общего дерева решений. При построении выберите подмножество функций в каждом узле и найдите лучший тест для одной из функций. Количество функций в выбранном подмножестве функций контролируется параметром max\_features. Чем меньше max\_features, тем больше различаются деревья в случайном лесу, но слишком маленький (когда 1) приведет к невозможности выбрать, какую функцию тестировать во время разделения. В sklearn max\_features имеет следующие методы выбора: «auto», «sqrt», «log2», None. И auto, и sqrt извлекают квадратный корень из общего количества функций, log2 логарифмирует общее количество функций, None делает max\_features прямым равным общему количеству функций, а значение max\_features по умолчанию - «auto».

Еще один важный параметр случайного леса - n\_jobs, который определяет количество используемых ядер ЦП.Использование большего количества ядер может увеличить скорость, а n\_jobs = -1 может вызывать все ядра. Конечно, max\_depth, min\_samples\_leaf и max\_leaf\_nodes также могут использоваться для предварительной обрезки во время строительства, чтобы уменьшить использование памяти и потребление времени.

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

from sklearn.ensemble import RandomForestClassifier

from sklearn.datasets import load\_breast\_cancer

cancer = load\_breast\_cancer()

X\_train, X\_test, y\_train, y\_test = train\_test\_split(cancer.data, cancer.target,

stratify=cancer.target, random\_state=4)

forest5 = RandomForestClassifier(n\_estimators=5, random\_state=0).fit(X\_train, y\_train)

print("n\_estimators=5")

print("Accuracy on training set: {:.3f}".format(forest5.score(X\_train, y\_train)))

print("Accuracy on test set: {:.3f}\n".format(forest5.score(X\_test, y\_test)))

forest100 = RandomForestClassifier(n\_estimators=100, random\_state=0).fit(X\_train, y\_train)

print("n\_estimators=100")

print("Accuracy on training set: {:.3f}".format(forest100.score(X\_train, y\_train)))

print("Accuracy on test set: {:.3f}".format(forest100.score(X\_test, y\_test)))

n\_estimators=5

Accuracy on training set: 0.998

Accuracy on test set: 0.923

n\_estimators=100

Accuracy on training set: 1.000

Accuracy on test set: 0.972

При прогнозировании, если это задача регрессии, прогнозируйте каждое дерево в лесу и принимайте среднее значение результатов в качестве прогнозируемого значения; если это задача классификации, прогнозируйте каждое дерево в лесу, и каждое дерево дает все возможные выходные данные Для каждой метки возьмите среднее значение вероятности, данной каждым деревом, и выберите метку с самой высокой вероятностью в качестве результата предсказания, что называется мягким голосованием.

Случайный лес обладает всеми преимуществами деревьев решений, и значение данного признака более надежно, чем значение, заданное одним деревом. Однако случайные леса плохо работают с разреженными данными с более высокими измерениями, а скорость обучения и прогнозирования ниже, чем у линейных моделей.

***3.В чем отличие GridSearch от RandomSearch?***

GridSearchCV

Поиск по сетке, поиск параметров, то есть в пределах указанного диапазона параметров, настройка параметров в последовательности в соответствии с длиной шага, использование настроенных параметров для обучения учащегося и поиск наиболее точного параметра в проверочном наборе из всех параметров. Цикл и процесс сравнения. GridSearchCV может гарантировать, что наиболее точные параметры будут найдены в указанном диапазоне параметров, но это также недостаток поиска по сетке. Он требует обхода всех возможных комбинаций параметров, что требует очень много времени в условиях больших наборов данных и нескольких параметров. . Как правило, используется последний метод поиска случайных параметров RandomizedSearchCV.

RandomSearch

Метод использования RandomizedSearchCV фактически такой же, как и GridSearchCV, но он заменяет GridSearchCV поиск параметров по сетке случайной выборкой в ​​пространстве параметров. Для параметров с непрерывными переменными RandomizedSearchCV будет рассматривать его как распределение. Выборка - это то, что не может выполнить поиск по сетке. Его способность поиска зависит от установленного параметра n\_iter.

***4.Как влияет стандартизация признаков на работу модели леса?***

from google.colab import drive

drive.mount('/content/drive')

***5.Что такое "важность признака"? Есть ли аналоги показателя в моделях линейной и логистической регрессии?***

Важность функции - это метод оценки входных характеристик прогнозной модели, который выявляет относительную важность каждой функции при прогнозировании. Оценки важности функций могут быть рассчитаны для проблем, связанных с прогнозируемыми значениями (называемыми регрессией), и проблем, связанных с прогнозируемыми метками категорий (называемых классификацией).