سنا ساری نوایی

شماره دانشجویی : 810199435

هوش مصنوعی

پروژه صفر

توضیحات : در این پروژه میخواهیم زنده ماندن مسافران حاضر در کشتی را بررسی نماییم. و با استفاده از اطلاعات به دست آمده بفهمیم مسافر در کشتی زنده میماند یا خیر.

سوال 1 : در این قسمت ابتدا داده ها را از فایل csv میخوانیم و یک dataframe از آن تشکیل میدهیم.

```
In [2]: import pandas as pd

        data = pd.read_csv("train.csv")
        train_dataframe = pd.DataFrame(data)
```

describe : با استفاده از این تابع میتوانیم اطلاعاتی درمورد داده هایمان دریافت کنیم؛ به عنوان مثال مقدار و تعداد، میانگین، انحراف معیار، مینیموم، چارک اول، چارک دوم، چارک سوم و مقدار ماکسیمم را دریابیم.

```
In [8]: train_dataframe.describe()
```

Out[8]:

	PassengerId	Survived	Pclass	Age	SibSp	Parch	Fare
count	891.000000	891.000000	891.000000	714.000000	891.000000	891.000000	891.000000
mean	446.000000	0.383838	2.308642	29.699118	0.523008	0.381594	32.204208
std	257.353842	0.486592	0.836071	14.526497	1.102743	0.806057	49.693429
min	1.000000	0.000000	1.000000	0.420000	0.000000	0.000000	0.000000
25%	223.500000	0.000000	2.000000	20.125000	0.000000	0.000000	7.910400
50%	446.000000	0.000000	3.000000	28.000000	0.000000	0.000000	14.454200
75%	668.500000	1.000000	3.000000	38.000000	1.000000	0.000000	31.000000
max	891.000000	1.000000	3.000000	80.000000	8.000000	6.000000	512.329200

tail : با استفاده از این تابع میتوانیم به مقادیر آخر داده دسترسی داشته باشیم. در حالت عادی 5 داده آخر را برمیگرداند اما اگر بخواهیم به تعداد دلخواه ما داده برگردند، کافیست درون تابع مقدار مورد نظر خود را بنویسیم به عنوان مثال برای به دست آوردن 7 داده پایانی باید بنویسیم:

train_dataframe.tail(7)

```
In [11]: train_dataframe.tail()
```

Out[11]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
886	887	0	2	Montvila, Rev. Juozas	male	27.0	0	0	211536	13.00	NaN	S
887	888	1	1	Graham, Miss. Margaret Edith	female	19.0	0	0	112053	30.00	B42	S
888	889	0	3	Johnston, Miss. Catherine Helen "Carrie"	female	NaN	1	2	W./C. 6607	23.45	NaN	S
889	890	1	1	Behr, Mr. Karl Howell	male	26.0	0	0	111369	30.00	C148	C
890	891	0	3	Dooley, Mr. Patrick	male	32.0	0	0	370376	7.75	NaN	Q

head : با استفاده از این تابع میتوانیم به مقادیر اول داده دسترسی داشته باشیم. در حالت عادی 5 داده اول را برمیگرداند اما اگر بخواهیم به تعداد دلخواه ما داده برگردند، کافیست درون تابع مقدار مورد نظر خود را بنویسیم به عنوان مثال برای به دست آوردن 7 داده اول باید بنویسیم:

train_dataframe.head(7)

```
In [12]: train_dataframe.head()
```

Out[12]:

	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

Info : با استفاده از این تابع میتوانیم اطلاعاتی درمورد dataframe داشته باشیم. این اطلاعات شامل تعداد ستون ها، انواع داده ستون ها، تعداد سلول های هر ستون و میزان استفاده از حافظه میباشد.

```
In [13]: train_dataframe.info()
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 891 entries, 0 to 890
Data columns (total 12 columns):
 #   Column       Non-Null Count  Dtype
---  ------       --------------  -----
 0   PassengerId  891 non-null    int64
 1   Survived     891 non-null    int64
 2   Pclass       891 non-null    int64
 3   Name         891 non-null    object
 4   Sex          891 non-null    object
 5   Age          714 non-null    float64
 6   SibSp        891 non-null    int64
 7   Parch        891 non-null    int64
 8   Ticket       891 non-null    object
 9   Fare         891 non-null    float64
 10  Cabin        204 non-null    object
 11  Embarked     889 non-null    object
dtypes: float64(2), int64(5), object(5)
memory usage: 83.7+ KB
```

سوال 2 : همانطور که در قسمت قبل دیدیم، با استفاده از تابع info ، میتوانیم اطلاعاتی درمورد dataframe بدست بیاوریم که برخی از آنها دسته ای و برخی دیگر عددی هستند. حال میخواهیم برچسب گذاری کنیم. ستون sex را در نظر میگیریم؛ با استفاده از کد زیر اعداد بازه [0,1] را به ستون مورد نظر اختصاص میدهیم. astype('category').cat.codes در واقع داده های موجود در ستونی که انتخاب کردیم را به عدد اختصاص میدهد و اعداد مثل اندیس ها از 0 شروع میشود. و همانطور که در شکل پیداست، به مردان عدد 0 و به زنان عدد 1 اختصاص یافته است.

```
In [20]: train_dataframe["Sex"] = train_dataframe["Sex"].astype("category").cat.codes
         train_dataframe.head(12)
```

Out[20]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
0	1	0	3	Braund, Mr. Owen Harris	0	22.0	1	0	A/5 21171	7.2500	NaN	S
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	1	38.0	1	0	PC 17599	71.2833	C85	C
2	3	1	3	Heikkinen, Miss. Laina	1	26.0	0	0	STON/O2. 3101282	7.9250	NaN	S
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	1	35.0	1	0	113803	53.1000	C123	S
4	5	0	3	Allen, Mr. William Henry	0	35.0	0	0	373450	8.0500	NaN	S
5	6	0	3	Moran, Mr. James	0	NaN	0	0	330877	8.4583	NaN	Q
6	7	0	1	McCarthy, Mr. Timothy J	0	54.0	0	0	17463	51.8625	E46	S
7	8	0	3	Palsson, Master. Gosta Leonard	0	2.0	3	1	349909	21.0750	NaN	S
8	9	1	3	Johnson, Mrs. Oscar W (Elisabeth Vilhelmina Berg)	1	27.0	0	2	347742	11.1333	NaN	S
9	10	1	2	Nasser, Mrs. Nicholas (Adele Achem)	1	14.0	1	0	237736	30.0708	NaN	C
10	11	1	3	Sandstrom, Miss. Marguerite Rut	1	4.0	1	1	PP 9549	16.7000	G6	S
11	12	1	1	Bonnell, Miss. Elizabeth	1	58.0	0	0	113783	26.5500	C103	S

حالا به بررسی تابع info میپردازیم:

```
In [21]: train_dataframe.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 891 entries, 0 to 890
Data columns (total 12 columns):
 #   Column       Non-Null Count  Dtype
---  ------       --------------  -----
 0   PassengerId  891 non-null    int64
 1   Survived     891 non-null    int64
 2   Pclass       891 non-null    int64
 3   Name         891 non-null    object
 4   Sex          891 non-null    int8
 5   Age          714 non-null    float64
 6   SibSp        891 non-null    int64
 7   Parch        891 non-null    int64
 8   Ticket       891 non-null    object
 9   Fare         891 non-null    float64
 10  Cabin        204 non-null    object
 11  Embarked     889 non-null    object
dtypes: float64(2), int64(5), int8(1), object(4)
memory usage: 77.6+ KB
```

در قسمت قبل وقتی از info استفاده کرده بودیم، برای بخش Sex ، نوع داده را object نوشته بود ولی الان داده به صورت int8 تعریف شده است.

سوال 3 : با استفاده از تابع isna().sum() میتوان تعداد سطرهایی که در یک ستون مشخص مقادیر تعریف نشده دارند را پیدا کرد. برای تک تک ستون ها این عمل را انجام میدهیم و نتیجه به صورت زیر میباشد که مشاهده میکنیم ستون های Age و Cabin و Embarked دارای مقداری Nan میباشند:

```
In [22]: train_dataframe["PassengerId"].isna().sum()
```

Out[22]: 0

```
In [23]: train_dataframe["Survived"].isna().sum()
```

Out[23]: 0

```
In [24]: train_dataframe["Pclass"].isna().sum()
Out[24]: 0

In [25]: train_dataframe["Name"].isna().sum()
Out[25]: 0

In [26]: train_dataframe["Sex"].isna().sum()
Out[26]: 0

In [27]: train_dataframe["Age"].isna().sum()
Out[27]: 177

In [28]: train_dataframe["SibSp"].isna().sum()
Out[28]: 0

In [29]: train_dataframe["Parch"].isna().sum()
Out[29]: 0

In [30]: train_dataframe["Ticket"].isna().sum()
Out[30]: 0

In [31]: train_dataframe["Fare"].isna().sum()
Out[31]: 0

In [32]: train_dataframe["Cabin"].isna().sum()
Out[32]: 687

In [33]: train_dataframe["Embarked"].isna().sum()
Out[33]: 2
```

برای پر کردن این مقادیر، از dataframe.fillna() استفاده میکنیم. مقادیر تعریف نشده را با مقدار میانگین همان ستون جایگزین میکنیم.

```
In [39]: train_dataframe["Age"] = train_dataframe["Age"].fillna(train_dataframe["Age"].mean())
         train_dataframe["Age"].isna().sum()
Out[39]: 0
```

برای قسمت Age از این تکنیک استفاده کردیم و مقادیر خالی را با میانگین پر کردیم اما ستون Cabin مقادیر تعریف نشده آن بسیار زیاد است و باید آن را حذف کنیم. در مورد ستون Embarked ، چون مقادیر آن به صورت عددی نیست پس نمیتوان میانگین گرفت؛ از مد استفاده میکنیم برای پر کردن خانه های خالی و به این صورت عمل میکنیم که در جاهای خالی، بیشترین داده ای که در آن ستون تکرار شده است را قرار میدهیم.

```
In [47]: train_dataframe = train_dataframe.drop("Cabin", axis=1)
         train_dataframe.head()
Out[47]:
```

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

```
In [45]: train_dataframe["Embarked"] = train_dataframe["Embarked"].fillna(train_dataframe["Embarked"].mode().iat[0])
         train_dataframe["Embarked"].isna().sum()
Out[45]: 0
```

مزایا:

- روش میانگین گیری برای داده هایی با تعداد کم مناسب است و وقتی تعداد داده ها زیاد باشد، باعث کندی در اجرای برنامه میشود.
- میانگین گرفتن برای خانه های خالی ایده مناسبی است جهت اینکه ما یک اطلاعات جامع از داده هایمان داشته باشیم و این کار ما را به هدف نزدیک میکند.
- برای وقتی که دقت زیاد برایمان اهمیت نداشته باشد روش مناسبی میباشد.

معایب:

- این روش میانگین را تغییر نمیدهد ولی واریانس و انحراف معیار داده ها را تغییر میدهد و اطلاعات دقت قبلی خودشان را از دست میدهند.
- گاهی اوقات روش میانگین گیری جواب نمیدهد به عنوان مثال میخواهیم ستونی را پر کنیم که در آن تعداد دانش اموزان کلاسی قرار دارد و برخی داده ها را نداریم. با استفاده از این روش امکان دارد تعداد افراد حاضر در کلاس عدد اعشاری به دست بیاید که این منطقی نیست!
- ایراد دیگر این روش آن است که به عنوان مثال داده ای از افرادی داریم که در نظرسنجی شرکت کرده اند اما به برخی سوال ها جواب نداده اند. اگر ما داده های پر نشده را خودمان پر کنیم، در واقع صحت نظرسنجی را زیر سوال برده ایم!

سوال 4 : میخواهیم ستون هایی که مقادیر منحصر به فرد دارند را حذف کنیم. سه ستون مد نظر است. ستون PassengerId و Name و Ticket را حذف میکنیم زیرا بودنشان در حل مشکل ما تاثیری ندارد.

```
In [48]: train_dataframe = train_dataframe.drop("PassengerId", axis=1)
         train_dataframe = train_dataframe.drop("Name", axis=1)
         train_dataframe = train_dataframe.drop("Ticket", axis=1)
         train_dataframe.head()
```

Out[48]:

	Survived	Pclass	Sex	Age	SibSp	Parch	Fare	Embarked
0	0	3	0	22.0	1	0	7.2500	S
1	1	1	1	38.0	1	0	71.2833	C
2	1	3	1	26.0	0	0	7.9250	S
3	1	1	1	35.0	1	0	53.1000	S
4	0	3	0	35.0	0	0	8.0500	S

سوال 5 : برای به دست آوردن تعداد مسافران مرد و زن از تابع value_counts() استفاده میکنیم. قبلا مرد را به عدد 0 و زن را به عدد 1 تبدیل کرده بودیم. تعداد مسافر مرد 577 نفر و تعداد مسافر زن 314 نفر میباشد.

```
In [50]: train_dataframe["Sex"].value_counts()

Out[50]: 0    577
         1    314
         Name: Sex, dtype: int64
```

و حالا تعداد مسافران مرد را پیدا کنیم که در بندر مورد نظر سوار شده اند. برای اینکار ابتدا مسافران مرد و سپس بندر مورد نظر را در نظر میگیریم و سپس and استفاده میکنیم. جواب 441 میباشد.

```
In [69]: male = (train_dataframe["Sex"] == 0)
         Southampton = (train_dataframe["Embarked"] == "S")
         (male & Southampton).sum()

Out[69]: 441
```

سوال 6 : تمامی شرایط ذکر شده در صورت سوال را به صورت جداگانه بررسی میکنیم و در نهایت با استفاده از and جواب نهایی بدست می آید.

```
In [70]: over_35 = (train_dataframe["Age"] > 35)
         alone = ((train_dataframe["SibSp"] == 0) & (train_dataframe["Parch"] == 0))
         ticket = (train_dataframe["Pclass"] == 3)
         (over_35 & alone & ticket).sum()

Out[70]: 41
```

سوال 7 : از تابع loc() استفاده میکنیم. در واقع میتوان آن را یک جمله شرطی در نظر گرفت. مثلا کد قسمت پایین به این معناست که اگر مسافری در بندر Queenstown سوار شده است، آنگاه کرایه بلیط آن مسافر را در نظر بگیرد. در نهایت هم با استفاده از mean() میانگین را بدست می‌آوریم که مقدار 13.27602 را به ما نشان میدهد.

```
In [77]: import time
         first_start = time.time()
         print(train_dataframe.loc[(train_dataframe["Embarked"] == "Q"),"Fare"].mean())
         first_end = time.time()
         print (first_end - first_start)

         13.276029870129872
         0.002009868621826172
```

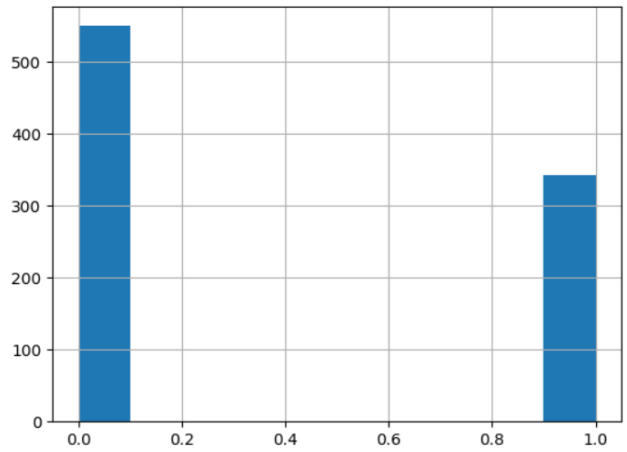
سوال 8 : کد سوال قبل را با استفاده از حلقه پیاده سازی کردیم و با استفاده از time.time() مقدار زمان را برای محاسبه زمان هر روش به دست آوردیم که همانطور که از این شکل و شکل قبلی پیداست، در این سوال سرعت کد کمتر از حالت قبل است و حدود 4 میلی ثانیه تفاوت دارند.

```
In [77]: import time
         first_start = time.time()
         print(train_dataframe.loc[(train_dataframe["Embarked"] == "Q"),"Fare"].mean())
         first_end = time.time()
         print (first_end - first_start)
```

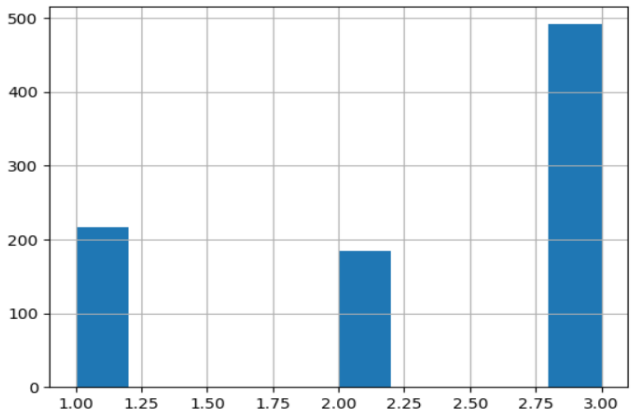
```
13.276029870129872
0.002009868621826172
```

سوال 9 :

```
In [83]: train_dataframe["Survived"].hist()
Out[83]: <AxesSubplot: >
```

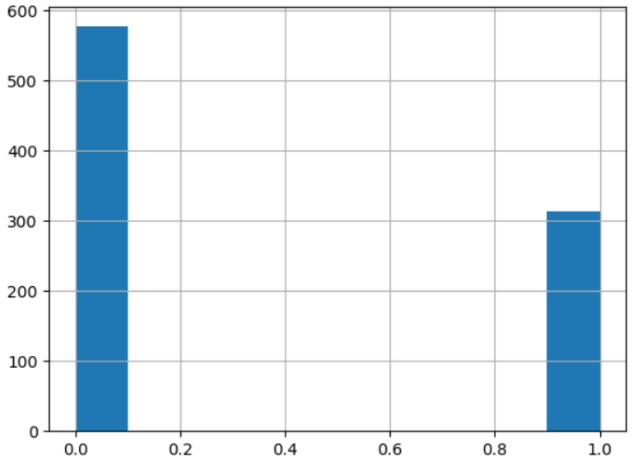



```
In [84]: train_dataframe["Pclass"].hist()
Out[84]: <AxesSubplot: >
```



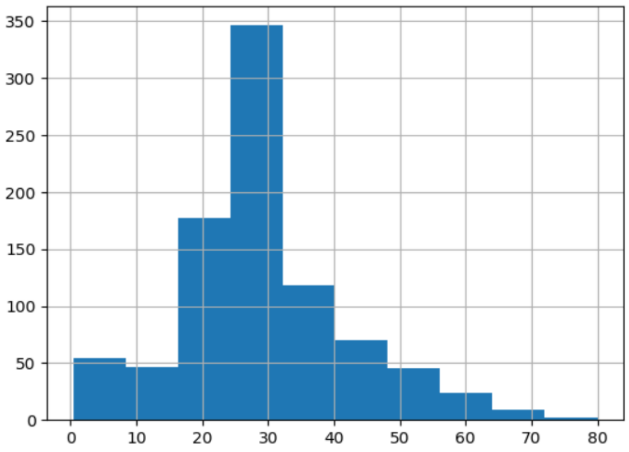
In [85]:
```
train_dataframe["Sex"].hist()
```

Out[85]: <AxesSubplot: >



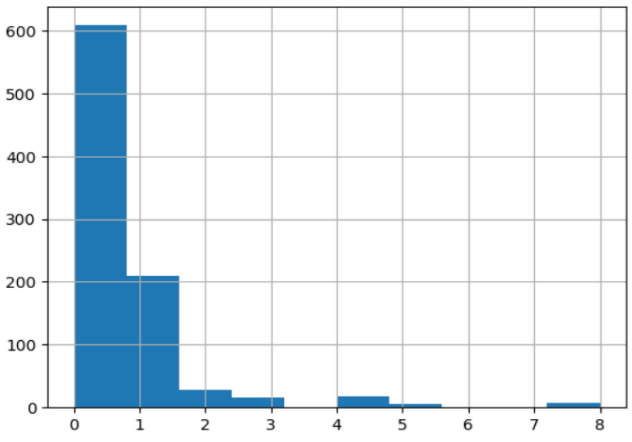
In [86]:
```
train_dataframe["Age"].hist()
```

Out[86]: <AxesSubplot: >



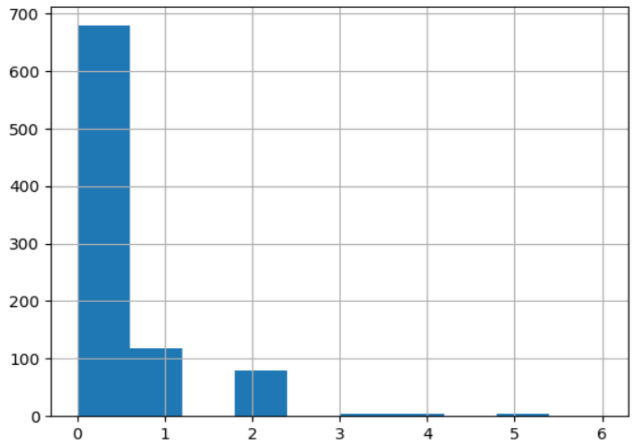
In [87]:
```
train_dataframe["SibSp"].hist()
```

Out[87]: <AxesSubplot: >



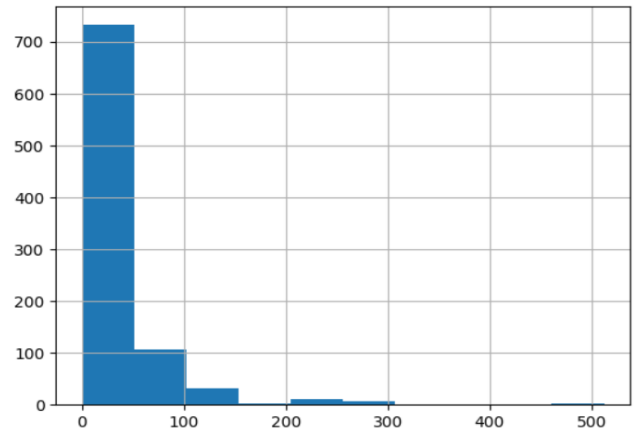
In [88]:
```
train_dataframe["Parch"].hist()
```

Out[88]: <AxesSubplot: >



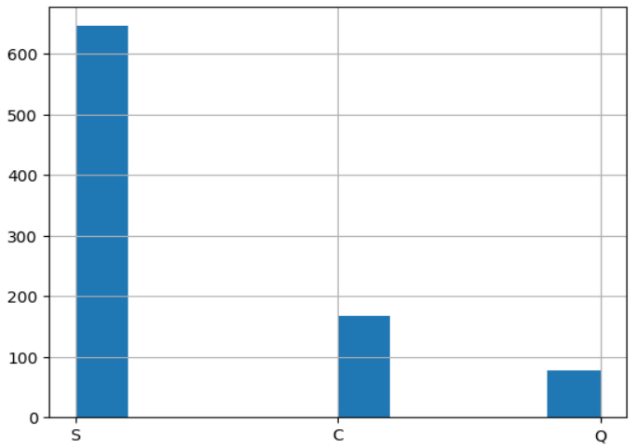
In [90]:
```
train_dataframe["Fare"].hist()
```

Out[90]: <AxesSubplot: >



In [89]:
```
train_dataframe["Embarked"].hist()
```

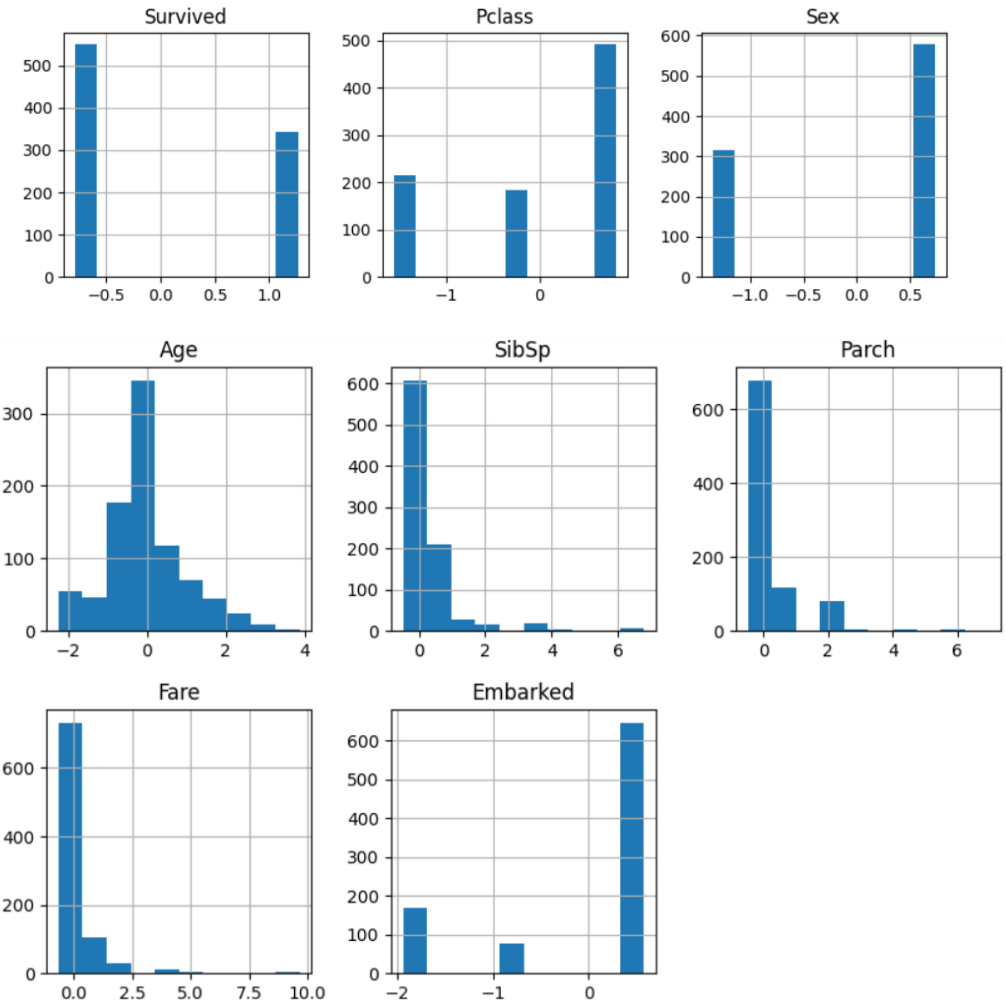
Out[89]: <AxesSubplot: >



سوال 10 : نتیجه نرمال سازی به صورت زیر میباشد:

```
In [38]: train_dataframe["Embarked"] = train_dataframe["Embarked"].astype("category").cat.codes
         columns = train_dataframe.select_dtypes(include=["number"]).columns
         train2 = (train_dataframe[columns] - train_dataframe[columns].mean()) / train_dataframe[columns].std()
         train2.hist(figsize=(10,10))
```

```
Out[38]: array([[<AxesSubplot: title={'center': 'Survived'}>,
                 <AxesSubplot: title={'center': 'Pclass'}>,
                 <AxesSubplot: title={'center': 'Sex'}>],
                [<AxesSubplot: title={'center': 'Age'}>,
                 <AxesSubplot: title={'center': 'SibSp'}>,
                 <AxesSubplot: title={'center': 'Parch'}>],
                [<AxesSubplot: title={'center': 'Fare'}>,
                 <AxesSubplot: title={'center': 'Embarked'}>, <AxesSubplot: >]],
               dtype=object)
```



سوال 11 : میانگین و انحراف معیار برای حالتی که مسافر زنده میماند:

In [63]:
```
col = train2.select_dtypes(include=["number"]).columns.drop("Survived")
Survived = train2.loc[train_dataframe["Survived"] == 1,col]
print("******mean*******\n",Survived.mean())
print("******std********\n",Survived.std())
```

```
******mean*******
 Pclass     -0.428611
Sex         -0.688034
Age         -0.088397
SibSp       -0.044728
Parch        0.103366
Fare         0.325822
Embarked    -0.212324
dtype: float64
******std********
 Pclass      1.032592
Sex          0.976297
Age          1.059259
SibSp        0.642659
Parch        0.957391
Fare         1.340157
Embarked     1.114980
dtype: float64
```

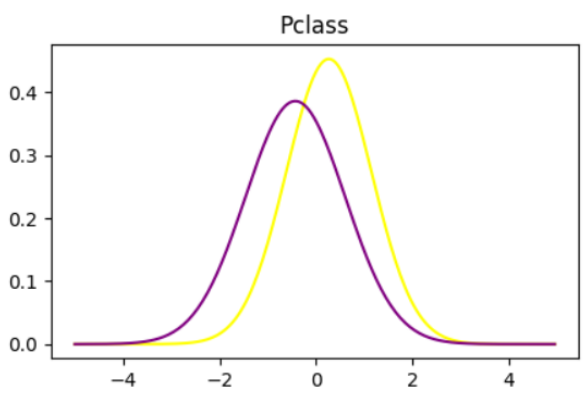
میانگین و انحراف معیار برای حالتی که مسافر زنده نمیماند:

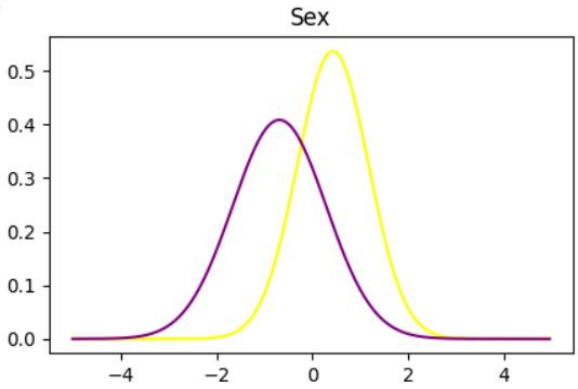
In [64]:
```
not_Survived = train2.loc[train_dataframe["Survived"] == 0,col]
print("******mean*******\n",not_Survived.mean())
print("******std********\n",not_Survived.std())
```

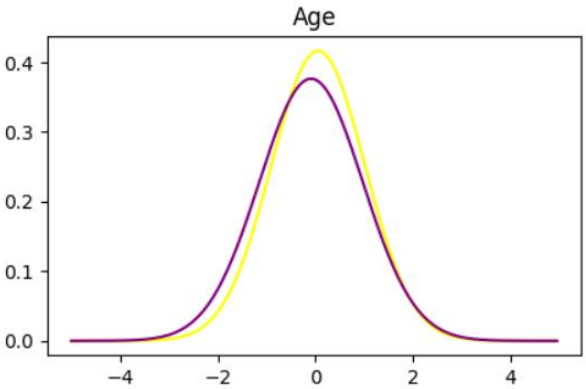
```
******mean*******
 Pclass      0.267004
Sex          0.428611
Age          0.055067
SibSp        0.027863
Parch       -0.064392
Fare        -0.202971
Embarked     0.132267
dtype: float64
******std********
 Pclass      0.880075
Sex          0.742626
Age          0.958111
SibSp        1.168358
Parch        1.021225
Fare         0.631637
Embarked     0.897130
dtype: float64
```

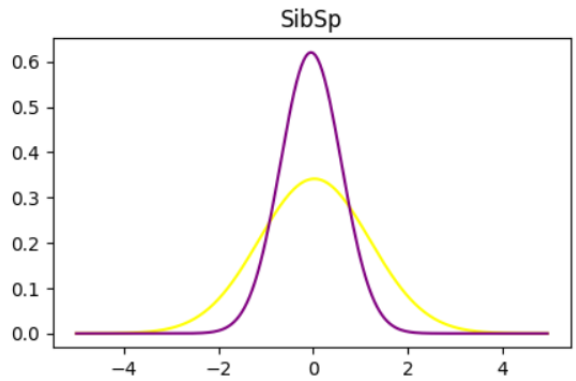
حال از این 4 داده یک dataframe درست میکنیم تا بتوانیم آنها را در یک plot بکشیم و مقایسه کنیم.

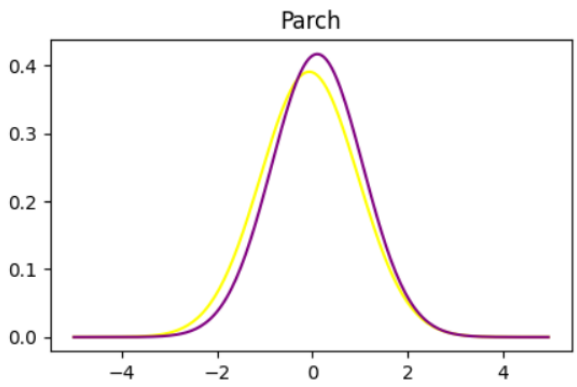
In [118]:
```
import matplotlib.pyplot as plt
from scipy.stats import norm

df = pd.DataFrame({"survived_mean":Survived.mean(),"survived_std": Survived.std(),"dead_mean":not_Survived.mean(),"dead_std":not_

def normal_shape(data):
    plt.figure(figsize=(5, 3))
    plt.plot(numpy.arange(-5, 5, 0.05),
        norm.pdf(numpy.arange(-5, 5, 0.05), loc=data["dead_mean"], scale=data["dead_std"]),color="yellow")
    plt.plot(numpy.arange(-5, 5, 0.05),
        norm.pdf(numpy.arange(-5, 5, 0.05), loc=data["survived_mean"], scale=data["survived_std"]),color="purple")
    plt.title(data.name)
    plt.show()

df.apply(normal_shape, axis=1);
```

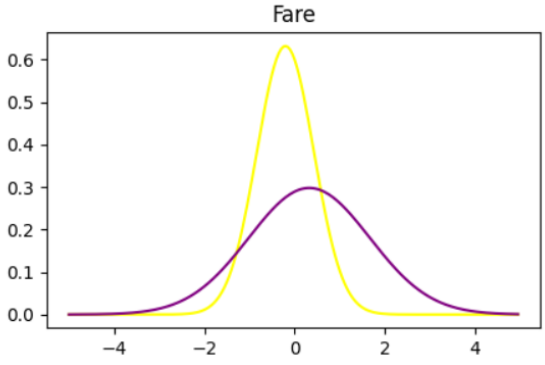

Pclass
0.4
0.3
0.2
0.1
0.0
−4
−2
0
2
4

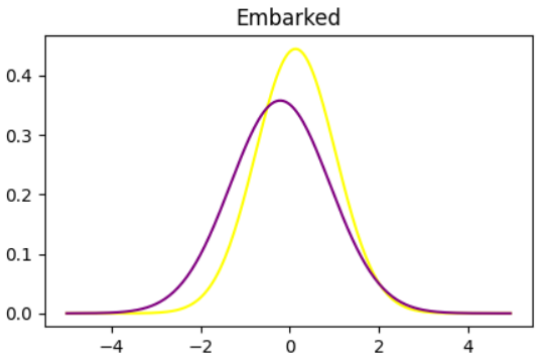

Sex
0.5
0.4
0.3
0.2
0.1
0.0
−4
−2
0
2
4


Age
0.4
0.3
0.2
0.1
0.0
−4
−2
0
2
4


SibSp
0.6
0.5
0.4
0.3
0.2
0.1
0.0
−4
−2
0
2
4


Parch
0.4
0.3
0.2
0.1
0.0
−4
−2
0
2
4


Fare
0.6
0.5
0.4
0.3
0.2
0.1
0.0
−4
−2
0
2
4


Embarked
0.4
0.3
0.2
0.1
0.0
−4
−2
0
2
4

حال میخواهیم نمودار ها را تحلیل کنیم و ببینیم کدام نمودار برای تشخیص زنده ماندن و نماندن مناسب تر است. نمودار هایی که دو داده آنها نزدیک به هم قرار دارند، یعنی آن ویژگی تاثیر چندانی بر زنده ماندن یا مردن مسافران ندارد پس از نظر ما آن نموداری بهتر است که اختلاف داده اول و دوم در شکل ملموس تر باشد و تفاوت آشکارتری داشته باشد. نمودار Fare بیشترین اختلاف را دارد با توجه به صفحه قبل، اما نمیتواند معیار مناسبی برای ما باشد زیرا قیمت بلیط هیچ تاثیری در زنده ماندن یا مردن افراد ندارد و مسیر های گوناگون قیمت های گوناگون دارند. اما نمودار های SibSp و Sex برای این کار مناسبند زیرا اختلاف آنها نیز از سایر نمودار ها بیشتر است و از نظر معنایی هم متناسب هستند و مثل Fare نیستند که هیچ تناسب معنایی با زنده ماندن یا نماندن نداشته باشند.

سوال 12 : کد سوال در زیر آمده است. بر اساس ویژگی Sex اقدام به زنده ماندن یا نماندن افراد کردیم زیرا نسبت به SibSp تغییرات آن بیشتر ملموس بود و در انتها در همان test.csv با استفاده از دستور to_csv آن را در فایل نوشتیم.

In [42]:
```
import pandas as pd
from scipy.stats import norm
import numpy as np
traindf = pd.read_csv("test.csv") ##reading
traindf["Sex"] = train_df["Sex"].astype("category").cat.codes

col_nums = traindf.select_dtypes(include=["number"]).columns
train3 = (traindf[col_nums] - traindf[col_nums].mean()) / traindf[col_nums].std() ##normalize

unsurvived_passenger = norm.pdf(train3["Sex"], df.loc["Sex"]["dead_mean"], df.loc["Sex"]["dead_std"])
survived_passenger = norm.pdf(train3["Sex"], df.loc["Sex"]["survived_mean"], df.loc["Sex"]["survived_std"])
traindf["Survived"] = np.where(unsurvived_passenger < survived_passenger , 1 , 0)

traindf.to_csv("test.csv", index = False) #writing
traindf
```
