```
import pandas as pd
from scipy import stats
import numpy as np
df= pd.read_csv("bmi.csv")
df.head()
```

```
{"summary":"{\n  \"name\": \"df\",\n  \"rows\": 500,\n  \"fields\": [\
n    {\n      \"column\": \"Gender\",\n      \"properties\": {\n
\"dtype\": \"category\",\n        \"num_unique_values\": 2,\n
\"samples\": [\n          \"Female\",\n          \"Male\"\n        ],\
n        \"semantic_type\": \"\",\n        \"description\": \"\"\n
}\n    },\n    {\n      \"column\": \"Height\",\n      \"properties\":
{\n        \"dtype\": \"number\",\n        \"std\": 16,\n
\"min\": 140,\n        \"max\": 199,\n        \"num_unique_values\":
60,\n        \"samples\": [\n          174,\n          147\
n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"Weight\",\n      \"properties\": {\n        \"dtype\": \"number\",\n
\"std\": 32,\n        \"min\": 50,\n        \"max\": 160,\n
\"num_unique_values\": 110,\n        \"samples\": [\n          124,\n
80\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"Index\",\n      \"properties\": {\n        \"dtype\": \"number\",\n
\"std\": 1,\n        \"min\": 0,\n        \"max\": 5,\n
\"num_unique_values\": 6,\n        \"samples\": [\n          4,\n
2\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    }\n  ]\
n}","type":"dataframe","variable_name":"df"}
```

```
df.dropna()
max_val= np.max(np.abs(df[['Height','Weight']]))
max_val
```

```
199
```

```
from sklearn.preprocessing import StandardScaler
sc= StandardScaler()
df1= df.copy()
df1[['Height','Weight']]= sc.fit_transform(df1[['Height','Weight']])
df1.head(10)
```

```
{"summary":"{\n  \"name\": \"df1\",\n  \"rows\": 500,\n  \"fields\":
[\n    {\n      \"column\": \"Gender\",\n      \"properties\": {\n
\"dtype\": \"category\",\n        \"num_unique_values\": 2,\n
\"samples\": [\n          \"Female\",\n          \"Male\"\n        ],\
n        \"semantic_type\": \"\",\n        \"description\": \"\"\n
}\n    },\n    {\n      \"column\": \"Height\",\n      \"properties\":
{\n        \"dtype\": \"number\",\n        \"std\":
1.001001502504384,\n        \"min\": -1.8304434706400559,\n
\"max\": 1.7761610166616852,\n        \"num_unique_values\": 60,\n
```

\"samples\": [\n   0.2479387762795916,\n   -1.4025412433330697\n   ],\n   \"semantic_type\": \"\",\n   \"description\": \"\"\n   }\n   },\n   {\n   \"column\": \"Weight\",\n   \"properties\": {\n   \"dtype\": \"number\",\n   \"std\": 1.0010015025043835,\n   \"min\": -1.7310552954326714,\n   \"max\": 1.6692318920243618,\n   \"num_unique_values\": 110,\n   \"samples\": [\n   0.5564106306747872,\n   -0.803704244308026\n   ],\n   \"semantic_type\": \"\",\n   \"description\": \"\"\n   }\n   },\n   {\n   \"column\": \"Index\",\n   \"properties\": {\n   \"dtype\": \"number\",\n   \"std\": 1,\n   \"min\": 0,\n   \"max\": 5,\n   \"num_unique_values\": 6,\n   \"samples\": [\n   4,\n   2\n   ],\n   \"semantic_type\": \"\",\n   \"description\": \"\"\n   }\n   }\n]\n}","type":"dataframe","variable_name":"df1"}

```
from sklearn.preprocessing import MinMaxScaler
mm= MinMaxScaler()
df2= df.copy()
df2[['Height','Weight']]= mm.fit_transform(df2[['Height','Weight']])
df2.head(10)
```

{"summary":"{\n \"name\": \"df2\",\n \"rows\": 500,\n \"fields\": [\n   {\n   \"column\": \"Gender\",\n   \"properties\": {\n   \"dtype\": \"category\",\n   \"num_unique_values\": 2,\n   \"samples\": [\n   \"Female\",\n   \"Male\"\n   ],\n   \"semantic_type\": \"\",\n   \"description\": \"\"\n   }\n   },\n   {\n   \"column\": \"Height\",\n   \"properties\": {\n   \"dtype\": \"number\",\n   \"std\": 0.27754679117955505,\n   \"min\": 0.0,\n   \"max\": 1.0,\n   \"num_unique_values\": 60,\n   \"samples\": [\n   0.5762711864406782,\n   0.1186440677966103\n   ],\n   \"semantic_type\": \"\",\n   \"description\": \"\"\n   }\n   },\n   {\n   \"column\": \"Weight\",\n   \"properties\": {\n   \"dtype\": \"number\",\n   \"std\": 0.2943873406331304,\n   \"min\": 0.0,\n   \"max\": 1.0,\n   \"num_unique_values\": 110,\n   \"samples\": [\n   0.6727272727272726,\n   0.27272727272727276\n   ],\n   \"semantic_type\": \"\",\n   \"description\": \"\"\n   }\n   },\n   {\n   \"column\": \"Index\",\n   \"properties\": {\n   \"dtype\": \"number\",\n   \"std\": 1,\n   \"min\": 0,\n   \"max\": 5,\n   \"num_unique_values\": 6,\n   \"samples\": [\n   4,\n   2\n   ],\n   \"semantic_type\": \"\",\n   \"description\": \"\"\n   }\n   }\n]\n}","type":"dataframe","variable_name":"df2"}

```
from sklearn.preprocessing import Normalizer
norm= Normalizer()
df3= df.copy()
```

```
df3[['Height','Weight']]= norm.fit_transform(df3[['Height','Weight']])
df3.head(10)
```


```
{"summary":"{\n  \"name\": \"df3\",\n  \"rows\": 500,\n  \"fields\":
[\n    {\n      \"column\": \"Gender\",\n      \"properties\": {\n
\"dtype\": \"category\",\n        \"num_unique_values\": 2,\n
\"samples\": [\n          \"Female\",\n          \"Male\"\n        ],\
n        \"semantic_type\": \"\",\n        \"description\": \"\"\n
}\n    },\n    {\n      \"column\": \"Height\",\n      \"properties\":
{\n        \"dtype\": \"number\",\n        \"std\":
0.07423492706686872,\n        \"min\": 0.6661089830104545,\n
\"max\": 0.9695636708716765,\n        \"num_unique_values\": 468,\n
\"samples\": [\n          0.8817846419346916,\n
0.7917822555634719\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"Weight\",\n      \"properties\": {\n        \"dtype\": \"number\",\n
\"std\": 0.12303393254888036,\n        \"min\": 0.24483931082618093,\n
\"max\": 0.7458544246384666,\n        \"num_unique_values\": 465,\n
\"samples\": [\n          0.47165225033716063,\n
0.56969552161906\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"Index\",\n      \"properties\": {\n        \"dtype\": \"number\",\n
\"std\": 1,\n        \"min\": 0,\n        \"max\": 5,\n
\"num_unique_values\": 6,\n        \"samples\": [\n          4,\n
2\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    }\n  ]\
n}","type":"dataframe","variable_name":"df3"}
```


```
from sklearn.preprocessing import MaxAbsScaler
mas= MaxAbsScaler()
df4= df.copy()
df4[['Height','Weight']]= mas.fit_transform(df4[['Height','Weight']])
df4.head(10)
```


```
{"summary":"{\n  \"name\": \"df4\",\n  \"rows\": 500,\n  \"fields\":
[\n    {\n      \"column\": \"Gender\",\n      \"properties\": {\n
\"dtype\": \"category\",\n        \"num_unique_values\": 2,\n
\"samples\": [\n          \"Female\",\n          \"Male\"\n        ],\
n        \"semantic_type\": \"\",\n        \"description\": \"\"\n
}\n    },\n    {\n      \"column\": \"Height\",\n      \"properties\":
{\n        \"dtype\": \"number\",\n        \"std\":
0.08228774210851135,\n        \"min\": 0.7035175879396985,\n
\"max\": 1.0,\n        \"num_unique_values\": 60,\n
\"samples\": [\n          0.8743718592964824,\n
0.7386934673366834\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"Weight\",\n      \"properties\": {\n        \"dtype\": \"number\",\n
\"std\": 0.2023912966852773,\n        \"min\": 0.3125,\n
\"max\": 1.0,\n        \"num_unique_values\": 110,\n
\"samples\": [\n          0.775,\n          0.5\n        ],\n
```

\"semantic_type\": \"\",\n   \"description\": \"\"\n   }\
n   },\n   {\n   \"column\": \"Index\",\n   \"properties\": {\
n   \"dtype\": \"number\",\n   \"std\": 1,\n   \"min\":
0,\n   \"max\": 5,\n   \"num_unique_values\": 6,\n
\"samples\": [\n   4,\n   2\n   ],\n
\"semantic_type\": \"\",\n   \"description\": \"\"\n   }\
n   }\n]\n}","type":"dataframe","variable_name":"df4"}

```
from sklearn.preprocessing import RobustScaler
rob= RobustScaler()
df5= df.copy()
df5[['Height','Weight']]= rob.fit_transform(df5[['Height','Weight']])
df5.head(10)
```

{"summary":"{\n \"name\": \"df5\",\n \"rows\": 500,\n \"fields\":
[\n   {\n   \"column\": \"Gender\",\n   \"properties\": {\n
\"dtype\": \"category\",\n   \"num_unique_values\": 2,\n
\"samples\": [\n   \"Female\",\n   \"Male\"\n   ],\
n   \"semantic_type\": \"\",\n   \"description\": \"\"\n
}\n   },\n   {\n   \"column\": \"Height\",\n   \"properties\":
{\n   \"dtype\": \"number\",\n   \"std\":
0.5848307385569197,\n   \"min\": -1.0892857142857142,\n
\"max\": 1.0178571428571428,\n   \"num_unique_values\": 60,\n
\"samples\": [\n   0.125,\n   -0.8392857142857143\n
],\n   \"semantic_type\": \"\",\n   \"description\": \"\"\n
}\n   },\n   {\n   \"column\": \"Weight\",\n   \"properties\":
{\n   \"dtype\": \"number\",\n   \"std\":
0.5782608476722207,\n   \"min\": -1.0,\n   \"max\":
0.9642857142857143,\n   \"num_unique_values\": 110,\n
\"samples\": [\n   0.32142857142857145,\n   -
0.4642857142857143\n   ],\n   \"semantic_type\": \"\",\n
\"description\": \"\"\n   }\n   },\n   {\n   \"column\":
\"Index\",\n   \"properties\": {\n   \"dtype\": \"number\",\n
\"std\": 1,\n   \"min\": 0,\n   \"max\": 5,\n
\"num_unique_values\": 6,\n   \"samples\": [\n   4,\n
2\n   ],\n   \"semantic_type\": \"\",\n
\"description\": \"\"\n   }\n   }\n]\
n}","type":"dataframe","variable_name":"df5"}

```
import pandas as pd
import numpy as np
import matplotlib
import matplotlib.pyplot as plt
import seaborn as sns
import statsmodels.api as sm
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.feature_selection import RFE
from sklearn.linear_model import RidgeCV, LassoCV, Ridge, Lasso
from sklearn.feature_selection import mutual_info_classif
```

```
from sklearn.feature_selection import mutual_info_regression
from sklearn.feature_selection import chi2

df= pd.read_csv("titanic_dataset.csv")
df.columns
```

```
Index(['PassengerId', 'Survived', 'Pclass', 'Name', 'Sex', 'Age',
'SibSp',
       'Parch', 'Ticket', 'Fare', 'Cabin', 'Embarked'],
      dtype='object')
```

```
df.shape
```

```
(891, 12)
```

```
x= df.drop("Survived",axis=1)
y=df['Survived']

df1= df.drop(['Name','Sex','Ticket','Cabin','Embarked'],axis=1)
df1.columns
```

```
Index(['PassengerId', 'Survived', 'Pclass', 'Age', 'SibSp', 'Parch',
'Fare'], dtype='object')
```

```
df1['Age'].isnull().sum()
```

```
177
```

```
df1['Age'].fillna(method='ffill')
```

```
<ipython-input-20-ad737854164f>:1: FutureWarning: Series.fillna with
'method' is deprecated and will raise in a future version. Use
obj.ffill() or obj.bfill() instead.
  df1['Age'].fillna(method='ffill')
```

```
0      22.0
1      38.0
2      26.0
3      35.0
4      35.0
       ...
886    27.0
887    19.0
888    19.0
889    26.0
890    32.0
Name: Age, Length: 891, dtype: float64
```

```
df1['Age']=df1['Age'].fillna(method='ffill')
```

```
<ipython-input-21-0051f5265196>:1: FutureWarning: Series.fillna with
'method' is deprecated and will raise in a future version. Use
```

```
obj.ffill() or obj.bfill() instead.
  df1['Age']=df1['Age'].fillna(method='ffill')

df1['Age'].isnull().sum()

0

from sklearn.feature_selection import SelectKBest,
mutual_info_regression
feature= SelectKBest(mutual_info_regression,k=3)
df1.columns

Index(['PassengerId', 'Survived', 'Pclass', 'Age', 'SibSp', 'Parch',
'Fare'], dtype='object')

cols= df1.columns.tolist()
cols[-1], cols[1] = cols[1], cols[-1]

df1= df1[cols]

df1.columns

Index(['PassengerId', 'Fare', 'Pclass', 'Age', 'SibSp', 'Parch',
'Survived'], dtype='object')

x= df1.iloc[:,0:6]
y= df1.iloc[:,6]
x.columns

Index(['PassengerId', 'Fare', 'Pclass', 'Age', 'SibSp', 'Parch'],
dtype='object')

y = y.to_frame()
y.columns

Index(['Survived'], dtype='object')

from sklearn.feature_selection import chi2
from sklearn.f
import pandas as pd

data= pd.read_csv("titanic_dataset.csv")
# Create the target variable 'y' before dropping the 'Survived' column
y= data['Survived']
data= data.drop(['Survived','Name','Ticket'], axis=1)
x

{"summary":"{\n  \"name\": \"x\",\n  \"rows\": 891,\n  \"fields\": [\n
{\n      \"column\": \"PassengerId\",\n      \"properties\": {\n
\"dtype\": \"number\",\n        \"std\": 257,\n        \"min\": 1,\n
\"max\": 891,\n        \"num_unique_values\": 891,\n
\"samples\": [\n          710,\n          440,\n          841\n
```

```
],\n        \"semantic_type\": \"\",\n        \"description\": \"\"\n
}\n    },\n    {\n      \"column\": \"Fare\",\n      \"properties\":
{\n        \"dtype\": \"number\",\n        \"std\":
49.693428597180905,\n        \"min\": 0.0,\n        \"max\":
512.3292,\n        \"num_unique_values\": 248,\n        \"samples\":
[\n          11.2417,\n          51.8625,\n          7.725\
n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"Pclass\",\n      \"properties\": {\n        \"dtype\": \"number\",\n
\"std\": 0,\n        \"min\": 1,\n        \"max\": 3,\n
\"num_unique_values\": 3,\n        \"samples\": [\n          3,\n
1,\n          2\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"Age\",\n      \"properties\": {\n        \"dtype\": \"number\",\n
\"std\": 14.554589813600963,\n        \"min\": 0.42,\n        \"max\":
80.0,\n        \"num_unique_values\": 88,\n        \"samples\": [\n
0.75,\n          22.0,\n          49.0\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"SibSp\",\n      \"properties\": {\
n        \"dtype\": \"number\",\n        \"std\": 1,\n        \"min\":
0,\n        \"max\": 8,\n        \"num_unique_values\": 7,\n
\"samples\": [\n          1,\n          0,\n          5\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"Parch\",\n      \"properties\": {\
n        \"dtype\": \"number\",\n        \"std\": 0,\n        \"min\":
0,\n        \"max\": 6,\n        \"num_unique_values\": 7,\n
\"samples\": [\n          0,\n          1,\n          4\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    }\n  ]\n}","type":"dataframe","variable_name":"x"}
```


```
data['Sex']= data["Sex"].astype("category")
data["Cabin"]= data["Cabin"].astype("category")
data["Embarked"]= data["Embarked"].astype("category")
data["Sex"]= data["Sex"].cat.codes
data["Cabin"]= data["Cabin"].cat.codes
data["Embarked"]= data["Embarked"].cat.codes
data
```


```
{"summary":"{\n  \"name\": \"data\",\n  \"rows\": 891,\n  \"fields\":
[\n    {\n      \"column\": \"PassengerId\",\n      \"properties\": {\
n        \"dtype\": \"number\",\n        \"std\": 257,\n
\"min\": 1,\n        \"max\": 891,\n        \"num_unique_values\":
891,\n        \"samples\": [\n          710,\n          440,\n
841\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"Pclass\",\n      \"properties\": {\n        \"dtype\": \"number\",\n
\"std\": 0,\n        \"min\": 1,\n        \"max\": 3,\n
\"num_unique_values\": 3,\n        \"samples\": [\n          3,\n
1,\n          2\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
```

\"Sex\",\n   \"properties\": {\n   \"dtype\": \"int8\",\n
\"num_unique_values\": 2,\n   \"samples\": [\n   0,\n
1\n   ],\n   \"semantic_type\": \"\",\n
\"description\": \"\"\n   }\n   },\n   {\n   \"column\":
\"Age\",\n   \"properties\": {\n   \"dtype\": \"number\",\n
\"std\": 14.526497332334044,\n   \"min\": 0.42,\n   \"max\":
80.0,\n   \"num_unique_values\": 88,\n   \"samples\": [\n
0.75,\n   22.0\n   ],\n   \"semantic_type\": \"\",\n
\"description\": \"\"\n   }\n   },\n   {\n   \"column\":
\"SibSp\",\n   \"properties\": {\n   \"dtype\": \"number\",\n
\"std\": 1,\n   \"min\": 0,\n   \"max\": 8,\n
\"num_unique_values\": 7,\n   \"samples\": [\n   1,\n
0\n   ],\n   \"semantic_type\": \"\",\n
\"description\": \"\"\n   }\n   },\n   {\n   \"column\":
\"Parch\",\n   \"properties\": {\n   \"dtype\": \"number\",\n
\"std\": 0,\n   \"min\": 0,\n   \"max\": 6,\n
\"num_unique_values\": 7,\n   \"samples\": [\n   0,\n
1\n   ],\n   \"semantic_type\": \"\",\n
\"description\": \"\"\n   }\n   },\n   {\n   \"column\":
\"Fare\",\n   \"properties\": {\n   \"dtype\": \"number\",\n
\"std\": 49.693428597180905,\n   \"min\": 0.0,\n   \"max\":
512.3292,\n   \"num_unique_values\": 248,\n   \"samples\":
[\n   11.2417,\n   51.8625\n   ],\n
\"semantic_type\": \"\",\n   \"description\": \"\"\n   }\
n   },\n   {\n   \"column\": \"Cabin\",\n   \"properties\": {\
n   \"dtype\": \"int16\",\n   \"num_unique_values\": 148,\n
\"samples\": [\n   15,\n   3\n   ],\n
\"semantic_type\": \"\",\n   \"description\": \"\"\n   }\
n   },\n   {\n   \"column\": \"Embarked\",\n   \"properties\":
{\n   \"dtype\": \"int8\",\n   \"num_unique_values\": 4,\n
\"samples\": [\n   0,\n   -1\n   ],\n
\"semantic_type\": \"\",\n   \"description\": \"\"\n   }\
n   }\n]\n}","type":"dataframe","variable_name":"data"}

```
from sklearn.feature_selection import chi2, SelectKBest # Import
SelectKBest
import pandas as pd

data= pd.read_csv("titanic_dataset.csv")
# Create the target variable 'y' before dropping the 'Survived' column
y= data['Survived']
data= data.drop(['Survived','Name','Ticket'], axis=1)
x
k=5
# Use SelectKBest instead of SelectkBest
selector= SelectKBest(score_func= chi2, k=k)
x_new= selector.fit_transform(x,y)
selected_feature_indices = selector.get_support(indices=True) #
Corrected the typo here
selected_features= x.columns[selected_feature_indices]
```

```
print("selected Features:")
print(selected_features)
```

```
selected Features:
Index(['PassengerId', 'Fare', 'Pclass', 'Age', 'Parch'],
dtype='object')
```

```
x.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 891 entries, 0 to 890
Data columns (total 6 columns):
 #   Column       Non-Null Count  Dtype
---  ------       --------------  -----
 0   PassengerId  891 non-null    int64
 1   Fare         891 non-null    float64
 2   Pclass       891 non-null    int64
 3   Age          891 non-null    float64
 4   SibSp        891 non-null    int64
 5   Parch        891 non-null    int64
dtypes: float64(2), int64(4)
memory usage: 41.9 KB
```

```
x= x.drop(["Name","Sex","Ticket","cabin","Embarked"],axis=1)
```

```
from sklearn.feature_selection import SelectKBest, f-regression
selector= SelectKBest(score_func= f_regression, k=5)
x_new= selector.fit_transform(x,y)
```

```
selected_feature_indices= selector.get_support(indices=True)
selected_features= x.columns[selected_feature_indices]
print("selected Features:")
print(selected_features)
```

```
selected Features:
Index(['PassengerId', 'Fare', 'Pclass', 'Age', 'Parch'],
dtype='object')
```

```
from sklearn.feature_selection import SelectKBest,
mutual_info_regression
selector= SelectKBest(score_func= mutual_info_regression, k=5)
x_new= selector.fit_transform(x,y)
```

```
selected_feature_indices= selector.get_support(indices=True)
selected_features= x.columns[selected_feature_indices]
print("selected Features:")
print(selected_features)
```

```
selected Features:
Index(['PassengerId', 'Fare', 'Pclass', 'Age', 'Parch'],
dtype='object')
```

```
from sklearn.feature_selection import SelectPercentile, chi2 #
Corrected the typo in module name from 'feauture_selection' to
'feature_selection'
selector= SelectPercentile(score_func= chi2, percentile=10) #
percentile is not a valid parameter for SelectKBest; you might want to
use SelectPercentile instead.
# If you want to select 10 features, use k=10 for SelectKBest
x_new= selector.fit_transform(x,y)

import pandas as pd
from sklearn.feature_selection import SelectFromModel
from sklearn.ensemble import RandomForestClassifier

model = RandomForestClassifier()
sfm= SelectFromModel(model, threshold='mean')
sfm.fit(x,y)
selected_features = x.columns[sfm.get_support()]
print("selected Features:")
print(selected_features)
```

```
selected Features:
Index(['PassengerId', 'Fare', 'Age'], dtype='object')
```

```
df= pd.read_csv("titanic_dataset.csv")
df.columns
```

```
Index(['PassengerId', 'Survived', 'Pclass', 'Name', 'Sex', 'Age',
'SibSp',
       'Parch', 'Ticket', 'Fare', 'Cabin', 'Embarked'],
      dtype='object')
```

```
df
```


```
{"summary":"{\n  \"name\": \"df\",\n  \"rows\": 891,\n  \"fields\": [\
n    {\n      \"column\": \"PassengerId\",\n      \"properties\": {\n
\"dtype\": \"number\",\n        \"std\": 257,\n        \"min\": 1,\n
\"max\": 891,\n        \"num_unique_values\": 891,\n
\"samples\": [\n          710,\n          440,\n          841\n
],\n        \"semantic_type\": \"\",\n        \"description\": \"\"\n
}\n    },\n    {\n      \"column\": \"Survived\",\n
\"properties\": {\n        \"dtype\": \"number\",\n        \"std\":
0,\n        \"min\": 0,\n        \"max\": 1,\n
\"num_unique_values\": 2,\n        \"samples\": [\n          1,\n
0\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"Pclass\",\n      \"properties\": {\n        \"dtype\": \"number\",\n
\"std\": 0,\n        \"min\": 1,\n        \"max\": 3,\n
\"num_unique_values\": 3,\n        \"samples\": [\n          3,\n
1\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"Name\",\n      \"properties\": {\n        \"dtype\": \"string\",\n
```

\"num_unique_values\": 891,\n   \"samples\": [\n
\"Moubarek, Master. Halim Gonios (\\\"William George\\\")\",\n
\"Kvillner, Mr. Johan Henrik Johannesson\"\n   ],\n
\"semantic_type\": \"\",\n   \"description\": \"\"\n   }\
n   },\n   {\n   \"column\": \"Sex\",\n   \"properties\": {\n
\"dtype\": \"category\",\n   \"num_unique_values\": 2,\n
\"samples\": [\n   \"female\",\n   \"male\"\n   ],\
n   \"semantic_type\": \"\",\n   \"description\": \"\"\n
}\n   },\n   {\n   \"column\": \"Age\",\n   \"properties\": {\
n   \"dtype\": \"number\",\n   \"std\": 14.526497332334044,\
n   \"min\": 0.42,\n   \"max\": 80.0,\n
\"num_unique_values\": 88,\n   \"samples\": [\n   0.75,\n
22.0\n   ],\n   \"semantic_type\": \"\",\n
\"description\": \"\"\n   }\n   },\n   {\n   \"column\":
\"SibSp\",\n   \"properties\": {\n   \"dtype\": \"number\",\n
\"std\": 1,\n   \"min\": 0,\n   \"max\": 8,\n
\"num_unique_values\": 7,\n   \"samples\": [\n   1,\n
0\n   ],\n   \"semantic_type\": \"\",\n
\"description\": \"\"\n   }\n   },\n   {\n   \"column\":
\"Parch\",\n   \"properties\": {\n   \"dtype\": \"number\",\n
\"std\": 0,\n   \"min\": 0,\n   \"max\": 6,\n
\"num_unique_values\": 7,\n   \"samples\": [\n   0,\n
1\n   ],\n   \"semantic_type\": \"\",\n
\"description\": \"\"\n   }\n   },\n   {\n   \"column\":
\"Ticket\",\n   \"properties\": {\n   \"dtype\": \"string\",\n
\"num_unique_values\": 681,\n   \"samples\": [\n
\"11774\",\n   \"248740\"\n   ],\n
\"semantic_type\": \"\",\n   \"description\": \"\"\n   }\
n   },\n   {\n   \"column\": \"Fare\",\n   \"properties\": {\n
\"dtype\": \"number\",\n   \"std\": 49.693428597180905,\n
\"min\": 0.0,\n   \"max\": 512.3292,\n
\"num_unique_values\": 248,\n   \"samples\": [\n
11.2417,\n   51.8625\n   ],\n   \"semantic_type\":
\"\",\n   \"description\": \"\"\n   }\n   },\n   {\n
\"column\": \"Cabin\",\n   \"properties\": {\n   \"dtype\":
\"category\",\n   \"num_unique_values\": 147,\n
\"samples\": [\n   \"D45\",\n   \"B49\"\n   ],\n
\"semantic_type\": \"\",\n   \"description\": \"\"\n   }\
n   },\n   {\n   \"column\": \"Embarked\",\n   \"properties\":
{\n   \"dtype\": \"category\",\n   \"num_unique_values\":
3,\n   \"samples\": [\n   \"S\",\n   \"C\"\n
],\n   \"semantic_type\": \"\",\n   \"description\": \"\"\n
}\n   }\n]\n}","type":"dataframe","variable_name":"df"}

```
import pandas as pd
from sklearn.feature_selection import SelectKBest, f_classif
from sklearn.impute import SimpleImputer # Import SimpleImputer for handling missing values

x = df[['PassengerId', 'Pclass', 'Age', 'SibSp', 'Parch', 'Fare']]
```

```
y = df['Survived']

# Create an imputer to replace NaN with the mean of the column
imputer = SimpleImputer(strategy='mean')

# Fit and transform the imputer on your feature data
x = imputer.fit_transform(x)

selector = SelectKBest(score_func=f_classif, k=4)
x_new = selector.fit_transform(x, y)

selected_feature_indices = selector.get_support(indices=True)
selected_features = df[['PassengerId', 'Pclass', 'Age', 'SibSp',
'Parch', 'Fare']].columns[selected_feature_indices] # Use original
columns for feature names
print("Selected Features:")
print(selected_features)
```

```
Selected Features:
Index(['Pclass', 'Age', 'Parch', 'Fare'], dtype='object')
```

```
import pandas as pd
import numpy as np
from scipy.stats import chi2_contingency # Corrected the function name

import seaborn as sns
tips= sns.load_dataset('tips')

tips.head()
```


```
{"summary":"{\n  \"name\": \"tips\",\n  \"rows\": 244,\n  \"fields\":
[\n    {\n      \"column\": \"total_bill\",\n      \"properties\": {\n
\"dtype\": \"number\",\n        \"std\": 8.902411954856856,\n
\"min\": 3.07,\n        \"max\": 50.81,\n
\"num_unique_values\": 229,\n        \"samples\": [\n          22.12,\
n          20.23,\n          14.78\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"tip\",\n      \"properties\": {\n
\"dtype\": \"number\",\n        \"std\": 1.3836381890011826,\n
\"min\": 1.0,\n        \"max\": 10.0,\n        \"num_unique_values\":
123,\n        \"samples\": [\n          3.35,\n          1.5,\n
6.73\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"sex\",\n      \"properties\": {\n        \"dtype\": \"category\",\n
\"num_unique_values\": 2,\n        \"samples\": [\n
\"Male\",\n          \"Female\"\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"smoker\",\n      \"properties\":
{\n        \"dtype\": \"category\",\n        \"num_unique_values\":
2,\n        \"samples\": [\n          \"Yes\",\n          \"No\"\n
],\n        \"semantic_type\": \"\",\n        \"description\": \"\"\n
```

```
}\n    },\n    {\n      \"column\": \"day\",\n      \"properties\": {\
n        \"dtype\": \"category\",\n        \"num_unique_values\": 4,\n
\"samples\": [\n          \"Sat\",\n          \"Fri\"\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"time\",\n      \"properties\": {\n
\"dtype\": \"category\",\n        \"num_unique_values\": 2,\n
\"samples\": [\n          \"Lunch\",\n          \"Dinner\"\
n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"size\",\n      \"properties\": {\n        \"dtype\": \"number\",\n
\"std\": 0,\n        \"min\": 1,\n        \"max\": 6,\n
\"num_unique_values\": 6,\n        \"samples\": [\n          2,\n
3\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    }\n  ]\
n}","type":"dataframe","variable_name":"tips"}
```