درس داده کاوی مجازی
زهرا احمدنژاد 400422011

```
# Number of unique classes in each object column
for col in dataset:
    x = dataset[col].unique()
    print(f"{col} -> unique values: {len(x)} ",sorted(x))
```

از این کد برای این استفاده میکنیم که ببینیم در هر کدام از این ستون ها چه مقادیر یکتایی وجود دارد. یعنی مقادیر یکتا با توجه به نوع فیچر به ما یک دیدی میدهد که هر فیچر چه مقادیری داشته و چه مقادیری میتوانست داشته باشد.

مثلا در feature 0 فقط عدد 0 و 1 وجود دارد.

feature 11 تنها یک مورد در این دیتاست دارد و هیچ داده ای به ما نمیدهد، بنابراین بی فایده است.

features 0,1,2,3,4,9,12,16,17,19 همه مقادیر را در محدوده خود دارند، بنابراین مقادیر احتمالاً نام مستعار برای نام کلاس آنها هستند. یعنی به جای اینکه اسم کلاس ها را در طبقه بندی بیاورد از این عددها استفاده کرده است.

features 5,6,7,8,10,13,14,15,18 احتمالاً برای مقادیر عددی سطل‌بندی شده‌اند.

```
# droping useless column
dataset.drop(['feature 11'], axis=1, inplace=True)
```

بنابراین feature 11 را چون بی فایده است جدا میکنیم.

```
for col in ['feature 0', 'feature 1', 'feature 2', 'feature 3', 'feature 4', 'feature 9', 'feature 12', 'feature
 16', 'feature 17', 'feature 19']:
    dataset[col] = dataset[col].astype('category')
```

فیچرهایی که نهایتا سه یا چهار مقدار داشتند را به عنوان category در نظر میگیریم. و بقیه فیچرها که مقادیر یکتای آنها بسیار زیاد بود را به عنوان numerical در نظر میگیریم.

```
numerical_column = dataset.select_dtypes(exclude="category").columns.tolist()
categorical_column = dataset.select_dtypes(include="category").columns.tolist()
print("Numerical Columns:", numerical_column)

print("**********")
print("Categorical Columns:", categorical_column)
```

برای مشخص کردن نوع فیچرها و هر ستون ازین کد استفاده میکنیم.

```
d = sns.pairplot(dataset, kind="scatter")
d.fig.set_size_inches(13,13)
plt.show()
```

اینجا دو تا دو تا فیچرها را با هم رسم کرده تا ببینیم چه مقادیری دارند. یعنی به ازازی مقادیر هر فیچر چه مقداری برای فیچر دیگر ثبت شده است. چه رابطه ای بین این دو فیچر وجود دارد. مثلا با بالا رفتن هرکدام دیگری چه تغییری میکند.

مثلا feature15 برای تمام مقادیر feature5 ، 8 بوده است.

```
# Categorical correlation too
associations(dataset, nominal_columns='auto', numerical_columns=None, mark_columns=Fals
e, nom_nom_assoc='cramer', num_num_assoc='pearson', ax=None, figsize=(15,10), annot=Tru
e, fmt='.2f', cmap=None, sv_color='silver', cbar=True, vmax=1.0, vmin=None, plot=True, comput
e_only=False, clustering=False, title=None, filename=None)
```

وابستگی بین فیچرها را نشان میدهد. هرچه عدد به دست آمده به 1+ نزدیکتر باشد، یعنی اینها بهم همبسته تر هستند و با افزایش هرکدام دیگری هم بالا میرود و هرچقدر به 1- نزدیکتر باشند با بالا رفتن یکی، دیگری پایین می آید. و هرچقدر به 0 نزدیکتر باشند یعنی هیچ وابستگی بهم ندارند.

tests

```
linregress(dataset['feature 13'], dataset['feature 14'])
```

linregress دو فیچر 13 و 14 را حساب کرده و p-value به دست آمده 3.9059097133753345e-13 است و چون این عدد خیلی کوچکتر از 0.05 است پس یک رابطه ای بین این دو فیچر وجود دارد.

```
for col1 in dataset:
  may_have_relation = []
  for col2 in dataset:
    if linregress(dataset[col1], dataset[col2]).pvalue < 0.05:
      may_have_relation.append(str(col2))
  may_have_relation.remove(col1)
  print(f"{col1} have relation with ->", may_have_relation)
```

یکی از ضرایبی که رگرسیون میدهد p-value است. p-value رابطه بین دو متغیر را حساب میکند. اگر p-value کمتر از 0.05 باشد اینها باهم رابطه دارند و اگر بیشتر باشد نمیتوانیم بگوییم باهم رابطه دارند و اگر خیلی بیشتر از 0.05 باشد این را کنار گذاشته و میگوییم بی فایده است.

حلقه ای که اینجا وجود دارد اینگونه است که میاد روی تک تک از این فیچرها با همه فیچرهای دیگر p-value را حساب میکند.

for col1 یعنی میرود اولین فیچر را برمیدارد و با همه فیچرهای دیگر linregress را حساب کرده و میگوید pvalue. اگر کمتر از 0.05 بود اینها را در یک آرایه بریز و بگو اینها باهم رابطه دارند.

```
# categorical test
for i in range(len(categorical_column)):
  for j in range(0,i):
    print()
    tmp = pd.crosstab(dataset[categorical_column[i]], dataset[categorical_column[j]])
    print(tmp)
    print()
    print(chi2_contingency(tmp))
    if (chi2_contingency(tmp)[1] < 0.05):
      print('there is a probable relation')
    print("\n-----------------")
```

این کد یک خلاصه ای از table می دهد.

مثلا میخواهیم ببینیم به ازای هرباری که feature0 = 0 بوده چندبار feature1 = 1 یا feature1 = 0 بوده است.

یعنی یک جدول درست میکند که خط بالا انگار ستونها اعداد feature0 هستند و سطرها اعداد ستون feature1 هستند.

میگوید به ازای باری که feature1 = 0 بوده و feature0 = 0 بوده، چندبار این اتفاق افتاده است، کل ستون را نگاه کرده و میبینیم 21 بار این اتفاق افتاده است وبه ازای همه این را محاسبه کرده است.

if (chi2_contingency(tmp)[1] < 0.05): این قسمت میگوید اگر این کمتر از 0.05 بود بگو اینها باهم رابطه دارند.

```
# categorical test
for i in range(len(categorical_column)):
  for j in range(0,i):
    if (chi2_contingency(pd.crosstab(dataset[categorical_column[i]], datas
et[categorical_column[j]]))[1] < 0.05):
      print(f'there is a probable relation betwenn {categorical_column[i]}
 and {categorical_column[j]}')
```

اینجا هم مانند قبلی است فقط جداکانه قسمت P-VALUE کد قبل را بررسی کرده است.