Feature Selection- Chi Square Test

Estimated time needed: **20** minutes

Objectives

After completing this lab you will have a good understanding in:

- F/P Values
- Chi Square Test
- Hypothesis Testing

This Notebook is created for Python Module

Fisher Score- Chisquare Test For Feature Selection

Compute chi-squared stats between each non-negative feature and class.

- This score should be used to evaluate categorical variables in a classification task.

This score can be used to select the n_features features with the highest values for the test chi-squared statistic from X, which must contain only non-negative features such as booleans or frequencies (e.g., term counts in document classification), relative to the classes.

Recall that the chi-square test measures dependence between stochastic variables, so using this function “weeds out” the features that are the most likely to be independent of class and therefore irrelevant for classification. The Chi Square statistic is commonly used for testing relationships between categorical variables.

It compares the observed distribution of the different classes of target Y among the different categories of the feature, against the expected distribution of the target classes, regardless of the feature categories.

In [1]:
```
import seaborn as sns
df=sns.load_dataset('titanic')
import numpy as np
import pandas as pd
```

In [2]:
```
DF = df.to_csv('titanic.csv', index = False)
```

In [3]:
```
df.sample(4)
```

Out[3]:

	survived	pclass	sex	age	sibsp	parch	fare	embarked	class	who	adult_male	deck	embark_town	alive	alone
703	0	3	male	25.0	0	0	7.7417	Q	Third	man	True	NaN	Queenstown	no	True
108	0	3	male	38.0	0	0	7.8958	S	Third	man	True	NaN	Southampton	no	True
663	0	3	male	36.0	0	0	7.4958	S	Third	man	True	NaN	Southampton	no	True
607	1	1	male	27.0	0	0	30.5000	S	First	man	True	NaN	Southampton	yes	True

In [6]:
```
##['sex','embarked','alone','pclass','Survived']
df=df[['sex','embarked','alone','pclass','survived']]
df.head()
```

Out[6]:

	sex	embarked	alone	pclass	survived
0	male	S	False	3	0
1	female	C	False	1	1
2	female	S	True	3	1
3	female	S	False	1	1
4	male	S	True	3	0

In [25]:
```
df['sex']=np.where(df['sex']=="male",1,0)
df.head()
```

Out[25]:

	survived	pclass	sex	age	sibsp	parch	fare	embarked	class	who	adult_male	deck	embark_town	alive	alone

	survived	pclass	sex	age	sibsp	parch	fare	embarked	class	who	adult_male	deck	embark_town	alive	alone
0	0	3	1	22.0	1	0	7.2500	S	Third	man	True	NaN	Southampton	no	False
1	1	1	0	38.0	1	0	71.2833	C	First	woman	False	C	Cherbourg	yes	False
2	1	3	0	26.0	0	0	7.9250	S	Third	woman	False	NaN	Southampton	yes	True
3	1	1	0	35.0	1	0	53.1000	S	First	woman	False	C	Southampton	yes	False

In [28]:
```
### Let's perform label encoding on sex column
import numpy as np
### let's perform label encoding on embarked
ordinal_label = {k: i for i, k in enumerate(df['embarked'].unique(), 0)}
df['embarked'] = df['embarked'].map(ordinal_label)
```

In [29]:
```
df.sample(10)
```

Out[29]:

	survived	pclass	sex	age	sibsp	parch	fare	embarked	class	who	adult_male	deck	embark_town	alive	alone
543	1	2	1	32.0	1	0	26.0000	0	Second	man	True	NaN	Southampton	yes	False
341	1	1	0	24.0	3	2	263.0000	0	First	woman	False	C	Southampton	yes	False
250	0	3	1	NaN	0	0	7.2500	0	Third	man	True	NaN	Southampton	no	True
643	1	3	1	NaN	0	0	56.4958	0	Third	man	True	NaN	Southampton	yes	True
402	0	3	0	21.0	1	0	9.8250	0	Third	woman	False	NaN	Southampton	no	False
747	1	2	0	30.0	0	0	13.0000	0	Second	woman	False	NaN	Southampton	yes	True
324	0	3	1	NaN	8	2	69.5500	0	Third	man	True	NaN	Southampton	no	False
711	0	1	1	NaN	0	0	26.5500	0	First	man	True	C	Southampton	no	True
337	1	1	0	41.0	0	0	134.5000	1	First	woman	False	E	Cherbourg	yes	True
500	0	3	1	17.0	0	0	8.6625	0	Third	man	True	NaN	Southampton	no	True

In [30]:
```
### let's perform label encoding on alone
df['alone']=np.where(df['alone']==True,1,0)
```

In [31]:
```
df.head()
```

Out[31]:

	survived	pclass	sex	age	sibsp	parch	fare	embarked	class	who	adult_male	deck	embark_town	alive	alone
0	0	3	1	22.0	1	0	7.2500	0	Third	man	True	NaN	Southampton	no	0
1	1	1	0	38.0	1	0	71.2833	1	First	woman	False	C	Cherbourg	yes	0
2	1	3	0	26.0	0	0	7.9250	0	Third	woman	False	NaN	Southampton	yes	1
3	1	1	0	35.0	1	0	53.1000	0	First	woman	False	C	Southampton	yes	0
4	0	3	1	35.0	0	0	8.0500	0	Third	man	True	NaN	Southampton	no	1

In [15]:
```
### train Test split is usually done to avaoid overfitting
from sklearn.model_selection import train_test_split
X_train,X_test,y_train,y_test=train_test_split(df[['sex','embarked','alone','pclass']],
                                               df['survived'],test_size=0.3,random_state=100)
```

In [16]:
```
X_train.head()
```

Out[16]:

	sex	embarked	alone	pclass
69	1	0	0	3
85	0	0	0	3
794	1	0	1	3
161	0	0	1	2
815	1	0	1	1

In [17]:
```
X_train['sex'].unique()
```

Out[17]:
```
array([1, 0])
```

In [18]:
```
X_train.isnull().sum()
```

Out[18]:
```
sex         0
embarked    0
alone       0
pclass      0
dtype: int64
```

In [19]:
```
## Perform chi2 test
### chi2 returns 2 values
### Fscore and the pvalue
from sklearn.feature_selection import chi2
f_p_values=chi2(X_train,y_train)
```

In [20]:
```
f_p_values
```

Out[20]:
```
(array([65.67929505,  7.55053653, 10.88471585, 21.97994154]),
 array([5.30603805e-16, 5.99922095e-03, 9.69610546e-04, 2.75514881e-06]))
```

In [21]:
```
import pandas as pd
p_values=pd.Series(f_p_values[1])
p_values.index=X_train.columns
p_values
```

Out[21]:
```
sex         5.306038e-16
embarked    5.999221e-03
alone       9.696105e-04
pclass      2.755149e-06
dtype: float64
```

In [22]:
```
p_values.sort_index(ascending=False)
```

Out[22]:
```
sex         5.306038e-16
pclass      2.755149e-06
embarked    5.999221e-03
alone       9.696105e-04
dtype: float64
```

Observation

Sex Column is the most important column when compared to the output feature Survived

---

Great Job!

That's all we need to know for now! Congratulations, you have learnt one more topic and hands-on with Python.This Notebook is prepared by Sumit Kumar Shukla IBM ICE.

About Author

Mr. Sumit is a Subject Matter Expert at IBM, and a data Scientist with more than five years of experience tutoring students from IITs, NITs, IISc, IIMs, and other prestigious institutions. Google Data Studio certified and IBM certified data analyst Data Science, Machine Learning Models, Graph Databases, and Data Mining techniques for Predictive Modeling and Analytics, as well as data integration, require expertise in Machine Learning and programming languages such as Python, R, and Tableau.
