What is the Univariant test ?

- The elimination process aims to reduce the size of the input feature set and at the same time to retain the class discreminatory information for classification problems.In simple word unwanted feature are removed for getting the proper and input features and at the same time it we get the proper information for classification problem.
- In that we taking the analysis of the variance i.e.ANOVA is can be thought as an extension of the T-test
- The independent T-test is used to comapare the mean of the condition between two groups although this ANOVA test is based on F-Test.F-Test is any stastical test in which test stastic has an F-distributions under the null hypothesis.
- An F-Test is any stastical test in which the test stastic has an F-distribution under the null hypothesis.
- Analysis of variance (ANOVA) is collection of stastical models and their associated estimation procedures(such as the 'variation' among and between groups) used to analyze the diffrences among group means in a sample.
- In the T-test we compare the mean from the two groups but in ANOVA we compre the mean of the groups which two or more than two groups.
- F-test is used to comparing the factors of the total deviation.For example,in one-way or single factor ANOVA stastical significance is tested for by comparing the F-test stastic.
- The ANOVA was developed by stastacian Rounald Fisher that is also known as F-test . The ANOVA is based on the law of the total variance where the observed variance in particular variable particant into the attribute to the diffrent source of the variation.
- F = variance between the features/variance within the features
- We having the choices of ANOVA according to that we have to use it like f_classif,f_regression.

Classification :-

In [30]:
```
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import pandas as pd
import warnings
warnings.filterwarnings('ignore')
```

In [31]:
```
data= pd.read_csv('Churn_Modelling.csv')
```

In [32]:
```
data.head()
```

Out[32]:

	RowNumber	CustomerId	Surname	CreditScore	Geography	Gender	Age	Tenure	Balance	NumOfProducts	HasCrCard	IsActiveMember
0	1	15634602	Hargrave	619	France	Female	42	2	0.00	1	1	1
1	2	15647311	Hill	608	Spain	Female	41	1	83807.86	1	0	1
2	3	15619304	Onio	502	France	Female	42	8	159660.80	3	1	0
3	4	15701354	Boni	699	France	Female	39	1	0.00	2	0	0
4	5	15737888	Mitchell	850	Spain	Female	43	2	125510.82	1	1	1

In [33]:
```
data.tail()
```

Out[33]:

	RowNumber	CustomerId	Surname	CreditScore	Geography	Gender	Age	Tenure	Balance	NumOfProducts	HasCrCard	IsActiveMe
9995	9996	15606229	Obijiaku	771	France	Male	39	5	0.00	2	1	
9996	9997	15569892	Johnstone	516	France	Male	35	10	57369.61	1	1	
9997	9998	15584532	Liu	709	France	Female	36	7	0.00	1	0	
9998	9999	15682355	Sabbatini	772	Germany	Male	42	3	75075.31	2	1	
9999	10000	15628319	Walker	792	France	Female	28	4	130142.79	1	1	

In [34]:
```
data.isnull().sum()
```

Out[34]:
```
RowNumber          0
CustomerId         0
Surname            0
CreditScore        0
Geography          0
Gender             0
Age                0
Tenure             0
Balance            0
NumOfProducts      0
HasCrCard          0
IsActiveMember     0
EstimatedSalary    0
Exited             0
dtype: int64
```

In [35]:
```
data.drop(columns=['Surname'],inplace=True)
```

In [36]:
```
data.head()
```

Out[36]:

	RowNumber	CustomerId	CreditScore	Geography	Gender	Age	Tenure	Balance	NumOfProducts	HasCrCard	IsActiveMember	Estimate
0	1	15634602	619	France	Female	42	2	0.00	1	1	1	10
1	2	15647311	608	Spain	Female	41	1	83807.86	1	0	1	11
2	3	15619304	502	France	Female	42	8	159660.80	3	1	0	11
3	4	15701354	699	France	Female	39	1	0.00	2	0	0	9
4	5	15737888	850	Spain	Female	43	2	125510.82	1	1	1	7

In [37]:
```
from sklearn.preprocessing import LabelEncoder
lable = LabelEncoder()
data['Gender']=lable.fit_transform(data['Gender'])
```

In [38]:
```
lable = LabelEncoder()
data['Geography']=lable.fit_transform(data['Geography'])
```

In [39]:
```
data.drop(columns=['RowNumber','CustomerId'],inplace=True)
```

In [40]:
```
from sklearn.metrics import accuracy_score,confusion_matrix,classification_report
from sklearn.preprocessing import StandardScaler
from sklearn.ensemble import RandomForestClassifier
from sklearn.model_selection import train_test_split
from sklearn.feature_selection import VarianceThreshold,f_classif
```

In [41]:
```
x = data.drop(columns='Exited')
y = data['Exited']
```

In [42]:
```
x.shape,y.shape
```

Out[42]:
```
((10000, 10), (10000,))
```

In [43]:
```
x_train,x_test,y_train,y_test = train_test_split(x,y,train_size=0.80,stratify=y,random_state=42)
```

In [44]:
```
x_train.shape,x_test.shape
```

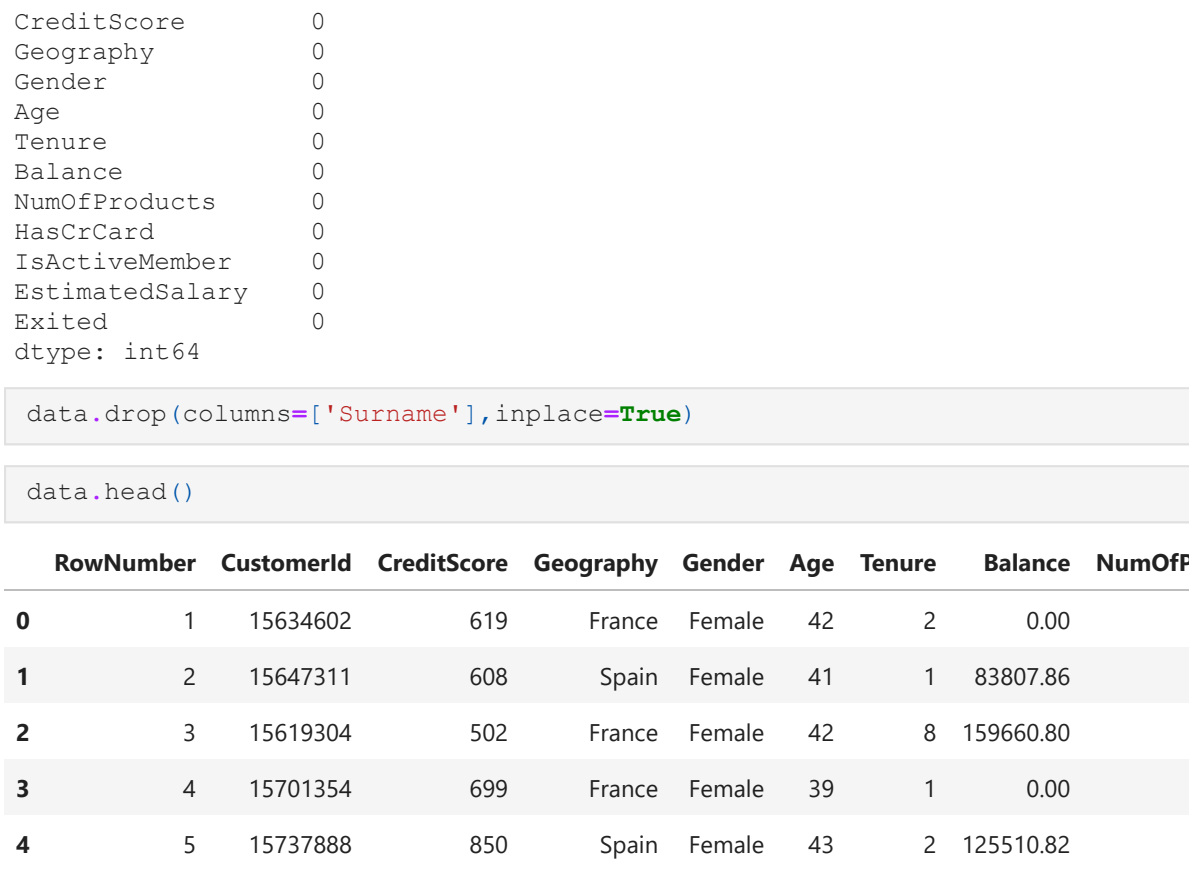
Out[44]:
```
((8000, 10), (2000, 10))
```

In [45]:
```
f_values,Pvalues = f_classif(x_train,y_train)
```

In [46]:
```
P_values = pd.Series(Pvalues,index=x_train.columns)
```

In [47]:
```
P_values.sort_values(ascending=True,inplace=True)
```

In [49]:
```
plt.figure(figsize=(10,6))
plt.bar(P_values.index,P_values,color='r')
plt.xticks(rotation=90)
plt.show()
```



Here whiever the feature have less p_value all are good to get the prediction.

In [ ]: