ESE Practical Examination 2021

DEPARTMENT: ARTIFICIAL INTELLIGENCE
SEM/SEC : 4th / A
SUBJECT : MACHINE LEARNING ALGORITHM
ROLL NO.: A-58
NAME: SHIVAM TAWARI
REG NO: 2019AAIE1117028

Title : Write a python program to evaluate a apply PCA Algorithm.

Introduction:

We have given a dataset on the Argentina provincial data. We need to apply PCA algorithm on it.
In the dataset we have the data of province, gdp, illiteracy, poverty, school dropout and etc.

PCA:

We are applying principal component analysis.

Principal component analysis is an unsupervised learning algorithm that is used for the dimensionality reduction in machine learning.

It is a statistical process that converts the observations of correlated features into a set of linearly uncorrelated features with the help of orthogonal transformation.

PCA generally tries to find the lower-dimensional surface to project the high-dimensional data.

Conclusion: Hence, successfully performed a program in python to apply PCA algorithm.

Code & Output:

PCA Algorithm on Principal Component Analysis on Argentina Provincial Data

```
# A-58 Shivam Tawari
# Import Packages
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.preprocessing import StandardScaler
from sklearn.decomposition import PCA
```

```
# Load Dataset
df = pd.read_csv('/content/argentina.csv')
df
```

	province	gdp	illiteracy	poverty	deficient_infra	school_dropout	no_healthcare	birth_mortal	pop
0	Buenos Aires	2.926899e+08	1.383240	8.167798	5.511856	0.766168	48.7947	4.4	15625084
1	Catamarca	6.150949e+06	2.344140	9.234095	10.464484	0.951963	45.0456	1.5	367828
2	Córdoba	6.936374e+07	2.714140	5.382380	10.436086	1.035056	45.7640	4.8	3308876
3	Corrientes	7.968013e+06	5.602420	12.747191	17.438858	3.864265	62.1103	5.9	992595
4	Chaco	9.832643e+06	7.517580	15.862619	31.479527	2.577462	65.5104	7.5	1055259

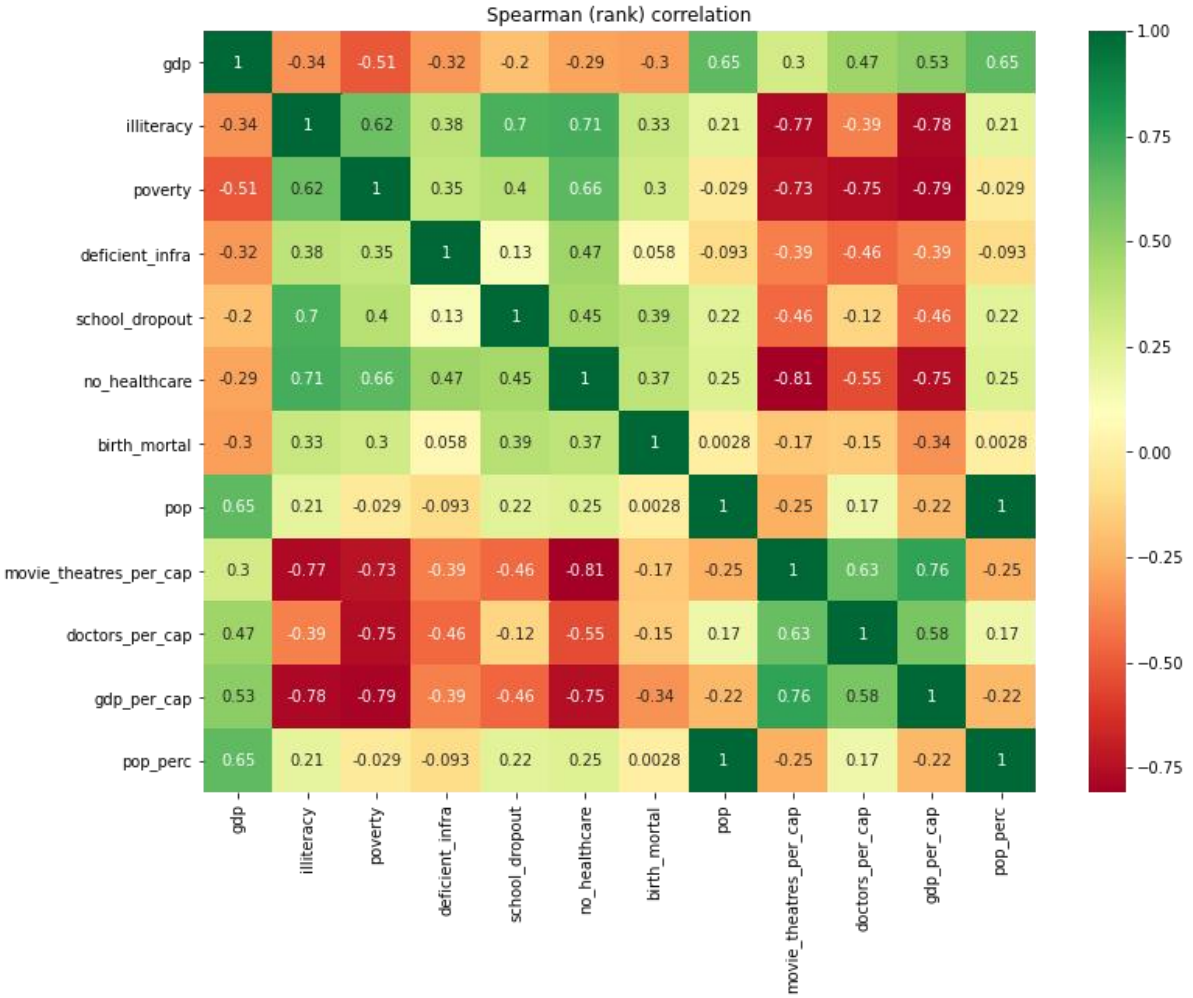
[3]
```
features_orig = ['gdp', 'illiteracy', 'poverty', 'deficient_infra',
                 'school_dropout', 'no_healthcare', 'birth_mortal', 'pop',
                 'movie_theatres_per_cap', 'doctors_per_cap']
df[features_orig].head()
```

	gdp	illiteracy	poverty	deficient_infra	school_dropout	no_healthcare	birth_mortal	pop	movie_theat
0	2.926899e+08	1.38324	8.167798	5.511856	0.766168	48.7947	4.4	15625084	
1	6.150949e+06	2.34414	9.234095	10.464484	0.951963	45.0456	1.5	367828	
2	6.936374e+07	2.71414	5.382380	10.436086	1.035056	45.7640	4.8	3308876	
3	7.968013e+06	5.60242	12.747191	17.438858	3.864265	62.1103	5.9	992595	
4	9.832643e+06	7.51758	15.862619	31.479527	2.577462	65.5104	7.5	1055259	

[4]
```
df['gdp_per_cap'] = np.round(df['gdp'] / df['pop'],3)
sum_pop = df['pop'].sum()
print('Overall population: ',sum_pop)
df['pop_perc'] = np.round(100 * df['pop'] / sum_pop,4)
features_new = ['gdp_per_cap', 'pop_perc']
features = features_orig + features_new
```

```
Overall population:  37099740
```

[5]
```
# RANK correlation of features
corr_mat = df[features].corr(method='spearman')
fig = plt.figure(figsize = (12,9))
sns.heatmap(corr_mat, annot=True, cmap="RdYlGn")
plt.title('Spearman (rank) correlation')
plt.show()
```




```
[6]  # Principal Component Analysis (PCA)
     featurespca = features.copy()
     featurespca.remove('pop_perc')
     featurespca.remove('gdp')
     print('Using the features:')
     print(featurespca)
```

```
Using the features:
['illiteracy', 'poverty', 'deficient_infra', 'school_dropout', 'no_healthcare', 'birth_mortal', 'pop', 'movie_theatres_per_
```

```
[7]  df4pca = df[featurespca]
     df4pca_std = StandardScaler().fit_transform(df4pca)
     pc_model = PCA(n_components=3)
     pc = pc_model.fit_transform(df4pca_std)
     df['pc_1'] = pc[:,0]
     df['pc_2'] = pc[:,1]
     df['pc_3'] = pc[:,2]
```

```
[8]  print(pc[:,0])
```

```
[-1.516474   -0.74411077 -2.48415692  2.80524741  4.29402088 -2.82022864
  0.01107419  4.18690575  0.76048044 -2.45357859 -0.05425972 -1.49877256
  2.99225537 -1.61392899 -1.00834045  1.90999894  0.22831465 -0.69953576
```

```
[8]  print(pc[:,0])
```

```
[-1.516474   -0.74411077 -2.48415692  2.80524741  4.29402088 -2.82022864
  0.01107419  4.18690575  0.76048044 -2.45357859 -0.05425972 -1.49877256
  2.99225537 -1.61392899 -1.00834045  1.90999894  0.22831465 -0.69953576
 -3.38446498 -1.55025833  2.69055205 -0.05073998]
```

```
[9]  print(pc[:,1])
```

```
[ 3.60070578  0.31412951  0.35485745  0.1402646  -0.58863385 -1.24966496
 -0.01268292 -1.50270374  0.53454055 -1.9282046  -1.37447024  0.48294814
  0.40436279 -0.74991948 -0.03590281  0.2318968   0.55554044  0.15275945
 -1.1434013   0.52219793  0.39210848  0.89927198]
```

```
[10] print(pc[:,2])
```

```
[ 0.40343249 -1.10009736  2.07078406  0.80775598 -0.32626332 -0.8083909
 -0.33055533  0.72024782 -1.34310577  0.30389488  2.25085056  0.27776396
  0.31775454 -0.63490156 -1.24209215 -0.78086546  0.86419152  0.34818807
 -1.25387313  1.01313519 -1.22434443 -0.33350967]
```