SDG 8 – DECENT WORK AND ECONOMIC GROWTH

Analysis of employment generation and contribution in economy by various factors

ES1101: COMPUTATIONAL DATA ANALYSIS

Prepared By: Team-2
Anay Sinhal (2021BTech012)
Babar Khilji (2021BTech031)
Diya Mathur (2021BTech041)
Kounen Aftab (2021BTech058)

TABLE OF CONTENT

LIST OF FIGURES

LIST OF TABLES

ABSTRACT

The main goal of this report is to analyse the different factors which is or can affect the employment generation or contribution in economy in India, these selected factors namely are government policies (MGNREGA), Start-ups, Micro small and medium enterprises, and tourism. Throughout this report we have learnt and applied the descriptive and inferential statistics, like mean, standard deviation, correlation, regression, test of hypothesis and ranking with the help of linear algebra. After applying the suitable methods to achieve purpose of this report, we came to know that these factors have generated employment or has a potential to do so. Also, they can boost our economy in the coming future. The government policy (MGNREGA) generated lot of employment opportunities since its implementation in 2006. Start-ups in India are affecting scope of employment and economy in different ways, and it can be a good factor to generate employment and boost economy in future. Micro, small, and medium enterprises contribute significantly to the economic growth of the country by fostering entrepreneurship and generating large employment opportunities at comparatively lower capital cost. Foreign tourist arrivals (tourism) play a major role in employment creation (direct and indirect employment).

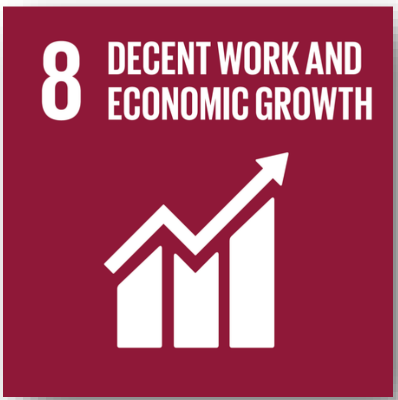
INTRODUCTION

What is SDG?

In 2015, United Nations decided some sustainable development goals (SDG) which are blueprint to achieve a better and more sustainable future for all. The SDGs were set up in 2015 by the United Nations General Assembly and are intended to be achieved by the year 2030. There are total 17 SDG:

Our team selected the SDG-8:

<u>SDG-8: Decent work and Economic Growth.</u>



SDG 8 has twelve goals to be achieved by 2030. Other goals are for 2030; some for 2020. The first ten are "outcome goals". These are: sustainable economic growth; diversity, innovation and improved economic productivity; promoting policies to support job creation and growing businesses; improve resource efficiency in use and production; full-time employment and decent work for equal pay; to promote youth employment, education and training; to abolish modern slavery, trafficking, and the exploitation of children; protect workers' rights and promote safe workplaces; to promote profitable and sustainable tourism; general access to banks, insurance and financial services. In addition, there are two "methods of achievement", namely: Increasing commercial support services; to develop a global youth employment strategy.

LITERATURE REVIEW

When we look at Target 3 of SDG-8 (i.e., Promote development-oriented policies that support productive activities, decent job creation, entrepreneurship, creativity, and innovation, and encourage the formalization and growth of micro-, small- and medium-sized enterprises, including through access to financial services.), we can clearly conclude that India have to work on several sectors in order to achieve the Target number-8.3 of SDG-8. A few of the factors India has started working on is decent job creation, entrepreneurship, creativity, and innovation, in one word we can say STARTUPS. After SDG (sustainable development goals) were decided in United Nations General Assembly in 2015. Since then, we have seen there is an enormous change in the number of start-ups in India. Prime Minister, Narendra Modi during his speech 5 August 2015 announced about the campaign “STARTUP INDIA”. It is an initiative to empower youth and the country. after reading some reports namely “THE ROLE OF STARTUPS FOR LOCAL LABOR MARKETS by Gerald Carlino”, “Measuring the Relationship between Start-ups Success and Job Creation by Dr. Shatakshee Dhongde” and **“**The Role of Entrepreneurship in US Job Creation and Economic Dynamism by Ryan Decker**”**, we felt that this can be a factor to generate & boost employment in India.

MGNREGA is the largest wage employment scheme ever launched in India, with 25% of rural households participating and the central government spending about 0.5% of GDP annually. Ghuman & Dua (2008) concluded that such a program would only be effective if there was sufficient social and political pressure from the local poor people on the executing agency. Dey (2010) concluded that further research is needed to know how the available resources will be used to create more jobs and to build up the useful rural infrastructure. Goswami (2013) observed that there have been significant evolutions in the implementation of MGNREGA using IT in all work phases. Rai (2010) and Jha (2012) stated that MGNREGA promised rural residents 100 days of work near their homes, but unfortunately, they weren’t provided with work within 15 days, as stipulated in the MGNREGA Act. He pointed out that no unemployment allowance was paid in that period. Anindita & Bhatia (2010) stated, in this scheme, 60% of the funds were used for wages, but although the construction was completed, the beneficiaries were not paid due to corruption or fraud. Pamela & Sharma (2015) observed that MGNREGA succeeded in improving rural household income. The authors observed that the implementation of MGNREGA reduced the movement of the study area.

MSME plays an important role in developing countries like India where millions of people are underemployed or unemployed. The aim of the report by (Udyog Aadhaar Memorandum 2019-2020) is to take an insight into untouched and neglected areas of MSME. The work of the government of India is an attempt at an initial study done on

Indian MSMEs under the manufacturing sector by taking several variables of Innovation under consideration. Report by S.L. Kapur Committee (2015) related to MSMEs are required to check the growth and analyse the performance of MSMEs and collect the data related to growth, while it is necessary that all the MSMEs must be registered on a common platform. The study on the growth and future prospectus of MSME units registered under Entrepreneurs Memorandum (EM-II) and Udyog Aadhar Memorandum (UAM) is hardly available. The Nayak Committee Report (2016) research exclusively emphasizes the growth and future prospects of registered units of MSMEs in India.

According to the analysis done in the report posted by the government **India Tourism Statistics (**the **year 2016, 2017, 2018, 2019 and 2020)**, it is observed that the total contribution of tourism to real GDP in India is exhibiting that its growth is been in a linear fashion over the past few years. The possible reason could be that importance is given to tourism by both Government and Private sectors. The major factor affecting it is the Foreign Tourist Arrivals. The increment in the tourist arrivals in the past years has also helped in changing to the current scenario where there's a lot of opportunities created for employment, for example- a taxi driver, a local tourist guide and many more. Also, there has been a lot of contribution to the different types of employment that are namely- Direct and Indirect Employment.

OBJECTIVES AND PROBLEM STATEMENTS

Objective 1: To Analyse Employment Generation by Government Policies.

PS1.1: To compare the number of persons registered vs employment provided to them under MGNREGA.
PS1.2: To study employment generated by MGNREGA and predict for future.
PS1.3: To determine the rank of states based on employment generated (per 10 lakh people)

Objective 2: To analyze how start-ups affect economy, employment, and its different aspects.

PS2.1: To analyze the effect of start-ups on the unemployment rate in India between 2016 and 2020
PS2.2: To analyze effect of start-ups on per capita GDP on India between the year 2016 and 2020.
PS2.3: To check how successful investments in start-ups are in India between the years 2016 and 2020.

Objective 3: To analyze how MSMEs affect Economy and Employment.

PS3.1: To analyze employment generated by MSMEs in India.
PS3.2: To analyze the share of MSMEs in the GDP.
PS3.3: To compare number of registrations of manufacturing enterprises vs services enterprises.

Objective 4: To study employment opportunities created by the tourism industries.

PS4.1: To analyze how Tourism is affecting employment in India.
PS4.2: To study the contribution of the tourism sector in the GDP.
PS4.3: To compare direct and indirect employment due to Tourism.

DATA COLLECTED

Objective 1: To Analyse Employment Generation by Government Policies.

PS1.1: To compare the number of persons registered vs employment provided to them under MGNREGA.

PS1.2: To study employment generated by MGNREGA and predict for future.

PS1.3: To determine the rank of states based on employment generated (per 10 lakh people)

Table 1.1 : Persons Registered Under MGNREGA

Source: MGNREGA Public Data Portal

States	2014	2015	2016	2017	2018	2019	2020	2021
AN	77753	78165	56537	54416	54096	52464	51521	50817
AP	19407928	19632776	18157986	1.8E+07	1.8E+07	1.8E+07	19366351	19538854
AR	491975	504159	495507	471181	459893	468247	490392	496511
AS	8182586	8860491	8260338	8191078	8597682	9494541	10570214	10808934
BH	20151892	21142177	22519754	2.3E+07	2.4E+07	2.6E+07	29003484	31489754
CG	12849439	12118105	9815806	8971706	9114638	9674305	10180118	9959846
GA	45323	46346	46891	47588	47744	48011	49080	49582
GJ	8961333	8902129	8339536	8451629	8758592	9191460	9711741	9883006
HR	1564634	1497387	1608459	1632647	1703967	1823700	2060668	2125254
HP	2326182	2311557	2303419	2300947	2363729	2467481	2630967	2720325
JK	2492864	2581104	2409110	2223379	2269758	2369683	2311519	2291848
JH	8052384	8077563	8539488	8020400	8456202	9081303	11022193	11329680
KR	16670896	16158186	14299861	1.4E+07	1.4E+07	1.5E+07	17033509	17695923
KL	4907293	5139035	5117504	5238555	5497161	5736629	6102862	6322485
LD	17554	16651	16449	16181	16039	16090	16141	16171
MP	28004524	25829501	18580177	1.7E+07	1.7E+07	1.9E+07	20343625	16960616
MH	19776662	20180854	20735755	2.1E+07	2.2E+07	2.2E+07	24042826	26760948
MN	1231835	1242910	1144301	1041462	1058264	1095455	1084782	1068256
MG	1126891	1113626	1131140	1136932	1155022	1192714	1223941	1238455
MZ	414008	412456	399443	341023	241155	245755	232942	222820
NL	828014	830497	809113	754221	755001	733997	744983	748712
OD	18381816	18628953	17528972	1.6E+07	1.7E+07	1.8E+07	18913850	18210098
PD	163750	159650	158774	141218	144177	149238	154037	155477
PB	1979386	2083817	2206349	2467373	2667245	2890665	3182488	3259656
RJ	25429316	25100226	23723053	2.3E+07	2.4E+07	2.5E+07	26277489	26522667
SK	165625	159918	140017	134655	137728	142468	146108	145156
TN	13189906	13516933	12027484	1.2E+07	1.2E+07	1.3E+07	13395512	13345854
TR	1435759	1269320	1183190	1071541	1088834	1125320	1160886	1166350

UP	24452875	25582783	23295630	2.3E+07	2.4E+07	2.6E+07	31769716	33367668
UK	2130042	2145177	1899397	1758526	1801240	1941928	2109686	2128857
WB	29095845	29635952	30009994	2.7E+07	2.8E+07	3E+07	32524174	33750554

Table 1.2 : Persons Employed Under MGNREGA

Source: MGNREGA Public Data Portal

States	2014	2015	2016	2017	2018	2019	2020	2021
AN	16975	11459	14589	8397	7080	6924	9532	4695
AP	5574494	6093683	6584514	6511777	6911313	6501703	7980526	7457706
AR	141120	188449	221792	147899	169246	182809	243783	257871
AS	1220962	2172391	2328607	2603364	2458778	2771325	3573300	3285311
BH	1158480	1829638	2793763	2702189	3382403	3856654	5852597	3854141
CG	3258964	4116624	3998142	4277448	4568868	4544184	6017988	4871401
GA	7500	6068	6932	6624	1161	1751	4365	3354
GJ	917093	999795	1279594	1393801	1516992	1326225	1931891	1649367
HR	325944	260027	417918	396050	326861	364019	651129	511547
HP	576187	541983	690675	671046	736077	716133	891623	878597
JK	420604	920253	868042	1011224	981268	957977	1166067	871727
JH	1572453	1568003	2470142	1908209	1623544	1762919	3205086	2852343
KR	3006239	3026118	4415073	3890970	3934472	4108973	5669612	6014117
KL	1513149	1693253	1643558	1471107	1708184	1654129	1882654	1774288
LD	493	135	8	142	259	124	75	23
MP	5823142	5178179	5202861	6097889	6673805	6169407	10538417	8752031
MH	2155396	2394554	2725353	3139461	3275876	2746113	3109983	2915444
MN	498833	484565	548329	527178	570353	590185	604233	584642
MG	463043	500274	545520	591429	679527	715056	752426	652665
MZ	261307	238102	212172	205269	200503	208685	223129	211928
NL	475483	533275	514652	453017	416321	425990	418827	440996
OD	2122586	3141545	3282338	3754645	3366339	3731602	6215596	5247267
PD	31671	35524	33563	41448	35931	43092	53406	35042
PB	337360	575774	650894	809495	824278	908128	1184255	1043251
RJ	5142506	6023326	6652485	6531324	7537391	8052322	11097709	9170940
SK	68014	81095	83261	74256	72178	67515	76985	70676
TN	6913415	7413183	7616367	6829325	6524708	6488785	7852828	7744962
TR	1099370	955865	929971	707277	739389	784279	833409	798628
UP	4692512	6834845	6225857	6039476	6150607	6453551	11654952	9104224
UK	541745	677766	704605	662234	638181	660855	909315	722018
WB	7348237	9154794	8528559	8101182	7442105	7968914	11826117	10365483

Objective 2: To analyze how start-ups affect economy, employment, and its different aspects.

PS2.1: To analyze the effect of start-ups on the unemployment rate in India between 2016 and 2020

Table 2.1 : Number of recognized start-ups

Source: Department for Promotion of Industry and Internal Trade

States	2016	2017	2018	2019	2020
Maharashtra	93	1091	1650	2179	2722
Karnataka	67	880	1208	1707	1761
Delhi	74	738	1185	1417	1801
Uttar Pradesh	29	410	788	896	1391
Gujarat	29	296	451	629	883
Haryana	28	267	486	720	817
Telangana	20	322	507	610	814
Tamil Nadu	54	268	459	621	767
Kerala	24	169	331	664	706
Rajasthan	14	139	245	354	503
West Bengal	8	177	275	314	404
Madhya Pradesh	7	106	297	335	428
Odisha	4	115	168	185	279
Andhra Pradesh	4	101	161	178	234
Bihar	1	48	147	155	265
Chhattisgarh	11	57	121	162	155
Jharkhand	2	35	88	89	165
Punjab	7	30	68	95	146
Uttarakhand	4	45	69	97	114
Assam	10	35	68	67	119
Jammu and Kashmir	2	15	47	38	64
Goa	2	19	44	41	67
Chandigarh	9	22	27	40	56
Himachal Pradesh		9	16	29	41
Puducherry		3	16	10	14
Manipur		4	7	6	12
Tripura			4	7	23
Andaman and Nicobar Islands		1	2	8	5
Nagaland	1	4	2	2	5
Dadra and Nagar Haveli and Daman and Diu		4	1	3	5
Meghalaya			2	5	
Arunachal Pradesh			2	2	

Mizoram			2	1	1
Sikkim		1		2	1
Ladakh					1
Lakshadweep					1
Total	**504**	**5411**	**8944**	**11668**	**14770**

Table2.2: Total Number of Start-ups and Unemployment in India

Source: World Bank

Calendar Year	Recognized Starts-ups	Unemployment Rate
2016	504	5.662
2017	5411	5.549
2018	8944	5.37
2019	11688	5.371
2020	14770	6.469

PS2.2: To analyze effect of start-ups on per capita GDP between the year 2016 and 2020.

Table 2.3: Number of Recognized Start-ups and GDP per Capita

Source: Macro trends

Calendar Year	Recognized Starts-ups	GDP per capita (in $)
2016	504	1733
2017	5411	1981
2018	8944	1997
2019	11688	2101
2020	14770	1901

PS2.3: To check how successful investments in start-ups are between the years 2016 and 2020.

Table 2.4 : Investment and Revenue Generated in Start-ups

Source: live mint and ADB org

YEAR	Recognized Starts-ups	Investment in start-ups (in billion USD)	Revenue Generated by start-ups (In billion USD)
2016	504	4.2	4.5
2017	5411	4.3	10.6
2018	8944	8.7	10.9
2019	11688	8.9	13

2020	14770	10.14	11.2

Objective 3: To analyze how MSMEs affect Economy and Employment.

PS3.1: To analyze employment generated by MSMEs in India.

Table 3.1 : Number of Jobs generated by MSMEs

Source: Udyog Aadhaar Memorandum

S. No.	State/UT	2015	2016	2017	2018	2019
1	ANDHRA PRADESH	136378	486416	276118	296131	250714
2	ARUNACHAL PRADESH	673	2705	3785	7598	4648
3	ASSAM	8738	16099	21847	41969	52896
4	BIHAR	238423	1505812	283105	293103	252242
5	CHHATTISGARH	44364	85277	55485	76692	93420
6	GOA	14457	17047	13641	17027	18901
7	GUJARAT	362417	1231337	1034520	1008251	793587
8	HARYANA	75321	381955	253632	388004	356651
9	HIMACHAL PRADESH	11351	35432	26027	47457	37982
10	JHARKHAND	54797	167448	126115	179179	146571
11	KARNATAKA	155662	567515	500585	694541	602530
12	KERALA	76744	185967	137645	183513	150197
13	MADHYA PRADESH	157638	364564	599041	1036168	771192
14	MAHARASHTRA	531121	1505936	1223733	2676784	2199867
15	MANIPUR	14949	54215	54498	59484	51890
16	MEGHALAYA	2	1987	3766	4518	2584
17	MIZORAM	3	4818	7359	7298	6491
18	NAGALAND	108	1862	1503	3488	4271
19	ODISHA	81000	222706	118972	137164	152755
20	PUNJAB	67906	277776	192576	317870	309468
21	RAJASTHAN	218457	583740	547159	580068	558365
22	SIKKIM	342	3716	2722	1522	1330
23	TAMIL NADU	301736	1918896	1387665	1540282	1217638
24	TELANGANA	173102	475478	374206	490761	349093
25	TRIPURA	4540	8675	5843	7555	6157
26	UTTAR PRADESH	280822	1777193	692268	779419	725334
27	UTTARAKHAND	19890	59073	45853	78151	77271
28	WEST BENGAL	149583	511755	255919	297912	289190
29	ANDAMAN AND NICOBAR ISLANDS	3361	7459	6415	6149	4698
30	CHANDIGARH	4030	19055	14938	24068	22944
31	DADAR AND NAGAR HAVELI	4859	12413	10617	19529	15993
32	DAMAN AND DIU	7959	12751	9674	22796	14545
33	DELHI	127789	333349	288896	457529	450473
34	JAMMU AND KASHMIR	786	12390	10680	21685	17474

35	LADAKH	0	0	0	0	0
36	LAKSHADWEEP	61	96	91	157	104
37	PUDUCHERRY	2575	25829	15105	18203	17681

PS3.2: To analyze the share of MSMEs in the GDP (in percentage).

Table 3.2 : Share of MSMEs in India's GDP (percent)

Source: Udyog Aadhaar Memorandum

S No.	State/UT	2015	2017	2017	2018	2019
1	ANDHRA PRADESH	1.97	6.81	3.59	1.81	2.78
2	ARUNACHAL PRADESH	0.01	0.01	0.01	0.02	0.02
3	ASSAM	0.01	0.04	0.11	0.2	0.37
4	BIHAR	19.94	23.26	6.53	3.42	2.73
5	CHHATTISGARH	0.96	0.27	0.54	0.71	1.2
6	GOA	0.16	0.05	0.08	0.1	0.12
7	GUJARAT	10.51	8.21	12.75	8.08	7.38
8	HARYANA	0.91	1	1.93	2.42	2.58
9	HIMACHAL PRADESH	0.19	0.08	0.12	0.18	0.22
10	JHARKHAND	4.23	2.09	1.41	1.21	1.25
11	KARNATAKA	2.86	2.01	3.72	4.01	4.42
12	KERALA	2.29	0.98	1.36	1.31	1.4
13	MADHYA PRADESH	7.88	3.67	13.58	14.12	12.01
14	MAHARASHTRA	10.92	8.83	12.57	26.79	27.02
15	MANIPUR	0.35	0.27	0.42	0.42	0.42
16	MEGHALAYA	0	0.01	0.04	0.03	0.02
17	MIZORAM	0	0.02	0.05	0.04	0.05
18	NAGALAND	0	0.01	0.01	0.02	0.03
19	ODISHA	1.73	1.73	1.17	0.68	1.05
20	PUNJAB	0.95	0.88	1.91	2.82	3.41
21	RAJASTHAN	6.84	4.3	8.1	5.94	6.89
22	SIKKIM	0.01	0	0.01	0.01	0.01
23	TAMIL NADU	8.41	11.27	14.36	10.98	10.03
24	TELANGANA	4.14	2.44	3.04	4.2	2.98
25	TRIPURA	0.1	0.05	0.05	0.07	0.06
26	UTTAR PRADESH	9.1	16.9	7.81	5.84	6.32
27	UTTARAKHAND	0.36	0.2	0.37	0.45	0.71
28	WEST BENGAL	3.31	3.44	2.4	1.48	1.62
29	ANDAMAN AND NICOBAR ISLANDS	0.1	0.04	0.1	0.07	0.06
30	CHANDIGARH	0.06	0.06	0.1	0.15	0.17
31	DADAR AND NAGAR HAVELI	0.05	0.03	0.05	0.07	0.06
32	DAMAN AND DIU	0.04	0.02	0.02	0.03	0.04
33	DELHI	1.53	0.84	1.45	2.12	2.33
34	JAMMU AND KASHMIR	0.01	0.11	0.07	0.11	0.1
35	LADAKH	0	0	0	0	0

36	LAKSHADWEEP	0	0	0	0	0
37	PUDUCHERRY	0.06	0.07	0.14	0.13	0.13

PS3.3: To compare number of registrations of manufacturing enterprises vs services enterprises.

Table 3.3 : Number of Registrations of Manufacturing vs Services Enterprises

Source: Udyog Aadhaar Memorandum

S no.	State/UT	2015			2016			2017			2018			2019		
		Manufacturing	Services	Total	Manufacturing	Services	Total	Manufacturing	Services	Total	Manufacturing	Services	Total	Manufacturing	Services	Total
1	ANDHRA PRADESH	6458	3270	9728	42248	119408	161656	27051	27466	54517	15054	23421	38475	20544	31257	51801
2	ARUNACHAL PRADESH	48	12	60	164	85	249	153	59	212	283	219	502	214	164	378
3	ASSAM	22	14	36	637	279	916	966	741	1707	1837	2376	4213	4034	2840	6874
4	BIHAR	30282	68442	98724	169542	382405	551947	45450	53714	99164	36799	35672	72471	16094	34841	50935
5	CHHATTISGARH	3728	1032	4760	4347	2165	6512	2618	5637	8255	3333	11748	15081	4376	18077	22453
6	GOA	562	220	782	766	380	1146	541	653	1194	784	1308	2092	659	1536	2195
7	GUJARAT	34426	17589	52015	116878	77965	194843	111584	81941	193525	92201	79124	171325	66732	70866	137598
8	HARYANA	3178	1338	4516	14903	8778	23681	13049	16283	29332	22385	28872	51257	19652	28505	48157
9	HIMACHAL PRADESH	629	315	944	1113	672	1785	996	793	1789	1821	1937	3758	1501	2644	4145
10	JHARKHAND	4871	16048	20919	18147	31354	49501	7732	13649	21381	9487	16233	25720	7221	16001	23222
11	KARNATAKA	9931	4223	14154	28891	18792	47683	29049	27470	56519	37238	47757	84995	31432	50876	82308
12	KERALA	7951	3361	11312	16236	6980	23216	12084	8483	20567	14328	13380	27708	12493	13663	26156
13	MADHYA PRADESH	15659	23352	39011	32151	54891	87042	70667	135398	206065	105571	193932	299503	72365	151420	223785
14	MAHARASHTRA	36463	17594	54057	93818	115629	209447	72999	117801	190800	162691	405592	568283	138060	365451	503511
15	MANIPUR	992	720	1712	3823	2702	6525	3797	2618	6415	5288	3568	8856	4065	3696	7761
16	MEGHALAYA	0	1	1	239	86	325	416	131	547	482	220	702	227	198	425
17	MIZORAM	0	1	1	277	108	385	601	198	799	437	318	755	471	388	859
18	NAGALAND	4	8	12	116	54	170	82	60	142	160	206	366	361	151	512
19	ODISHA	6470	2117	8587	9893	31178	41071	10498	7240	17738	6346	8072	14418	6324	13284	19608
20	PUNJAB	3929	773	4702	14533	6429	20962	12312	16705	29017	22061	37720	59781	23462	40081	63543
21	RAJASTHAN	19000	14882	33882	45327	56668	101995	49938	72970	122908	52030	73989	126019	47859	80515	128374
22	SIKKIM	28	21	49	50	65	115	84	87	171	46	208	254	41	135	176
23	TAMIL NADU	20714	20930	41644	125603	141888	267491	101500	116462	217962	108916	124008	232924	82762	104107	186869
24	TELANGANA	9085	11433	20518	26255	31605	57860	18249	27933	46182	23364	65699	89063	16450	38992	55442
25	TRIPURA	302	213	515	820	480	1300	326	373	699	822	729	1551	398	701	1099
26	UTTAR PRADESH	25323	19710	45033	194978	206049	401027	56815	61712	118527	54517	69382	123899	47990	69709	117699
27	UTTARAKHAND	1120	650	1770	2751	1970	4721	2446	3192	5638	3924	5549	9473	5987	7190	13177
28	WEST BENGAL	8241	8157	16398	36069	45475	81544	24214	12228	36442	17085	14210	31295	14035	16189	30224
29	ANDAMAN AND NICOBAR ISLANDS	218	300	518	345	713	1058	397	1059	1456	465	1022	1487	382	788	1170
30	CHANDIGARH	171	102	273	817	586	1403	545	1008	1553	914	2278	3192	885	2361	3246
31	DADAR AND NAGAR HAVELI	209	35	244	391	320	711	327	458	785	712	681	1393	573	577	1150
32	DAMAN AND DIU	177	2	179	357	79	436	258	121	379	410	263	673	375	297	672
33	DELHI	4589	2997	7586	10298	9749	20047	10009	11955	21964	20324	24604	44928	17400	26034	43434
34	JAMMU AND KASHMIR	49	9	58	395	2144	2539	362	628	990	927	1489	2416	844	1093	1937
35	LAKSHADWEEP	8	2	10	11	7	18	7	7	14	16	8	24	7	18	25
36	PUDUCHERRY	211	92	303	983	748	1731	965	1135	2100	1173	1520	2693	791	1665	2456
	Total	255048	239965	495013	1014172	1358886	2373058	689087	828368	1517455	824231	1297314	2121545	667066	1196310	1863376

Objective 4: To study employment opportunities created by the tourism industries.

PS4.1: To analyse how Tourism is affecting employment in India.

Table 4.1 : Number of Jobs generated by Tourism and Foreign Tourist Arrivals

Source: Ministry of Tourism

Years	Total Employees in the tourism sector (in Millions)	Foreign Tourist Arrivals (in Millions)
2014	**67.2**	7.8
2015	69.75	8.1
2016	72.26	8.8
2017	75.34	10.2
2018	80.63	10.6
2019	87.5	10.9

PS4.2: To study the contribution of the tourism sector in the GDP.

Table 4.2 : Share of Tourism sector into GDP

Source: Indian Brand Equity Foundation

Years	Total Contribution to GDP (in Billion USD)	Capital Investments in tourism Sector (in Billion USD)
2014	120.6	35
2015	129.5	34.6
2016	219.7	36.6
2017	232	47.8
2018	247.4	48.4
2019	191.3	42.3
2020	121.9	49

PS4.3: To compare direct and indirect employment due to Tourism.

Table 4.3 : Share of Direct and Indirect Employment on Tourism Sector

Source: Annual Report Ministry of Tourism 2021

Years	2013-14	2014-15	2015-16	2016-17	2017-18	2018-19
Shares in Jobs (%)	11.91	12.14	12.38	12.2	12.29	13
Indirect (%)	5.19	5.3	5.4	5.32	5.36	6
Direct (%)	6.72	6.84	6.98	6.88	6.93	7
Direct + Indirect (in Millions)	67.19	69.56	72.26	75.71	80.54	89

PROPOSED METHODOLOGIES

Descriptive Statistics:

Mean: The mean is the sum of all observations divided by the number of observations.

Median: The median is the middle value in an ordered data set.

Standard Deviation:

$$\sigma = \sqrt{\frac{\Sigma(x - \overline{x})^2}{n - 1}}$$

Correlation and Regression:

The most used techniques for investigating the relationship between two quantitative variables are correlation and linear regression. Correlation quantifies the strength of the linear relationship between a pair of variables, whereas regression expresses the relationship in the form of an equation.

Correlation between the variables is found using the correlation coefficient, *r*. The value of this coefficient is then measured on the scale of -1 to 1.

$$r = \frac{\sum (x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum (x_i - \bar{x})^2 \sum (y_i - \bar{y})^2}}$$

r = correlation coefficient
x = values of the x-variable in a population
x_i = avg. of the values of the x-variable
y_i= avg. of the y-variable in a population.
y = avg. of the values of the y-variable.

Pearson Correlation Coefficient:

$$r = \frac{\frac{1}{n}\Sigma xy - \overline{x}\overline{y}}{s_x s_y}$$

where $s_x = \sqrt{\frac{1}{n}\Sigma x^2 - \bar{x}^2}$ and $s_y = \sqrt{\frac{1}{n}\Sigma y^2 - \bar{y}^2}$.

Spearman's rank correlation coefficient:

$$r_s = 1 - \frac{6\Sigma d^2}{n\left(n^2 - 1\right)}$$

where $d = x - y$, is the difference in ranking.

Regression

$$Y_i = f(X_i, \beta) + e_i$$

$$a = \frac{(\Sigma y)(\Sigma x^2) - (\Sigma x)(\Sigma xy)}{n(\Sigma x^2) - (\Sigma x)^2}$$

$$b = \frac{n(\Sigma xy) - (\Sigma x)(\Sigma y)}{n(\Sigma x^2) - (\Sigma x)^2}$$

The equation has the form y = a + bx
Y is the dependent value.
X is the independent value.
β is the slope pf straight line
a is the y-intercept slope.

Power Method:

This method is used to find the dominant eigenvalue and a corresponding eigenvector. To apply this method on a matrix A, start with the initial guess for the eigenvector of dominant eigenvalue. Multiply the obtained vector on the left by A, bring this result in standard form, then repeat this process until the desired eigenvector is not found or until the results are not similar. If it occurs, then the norm of that eigenvector will be considered as the absolute value of the dominant eigenvalue.

Hypothesis Testing:

It is the method used in making statistical decisions using experimental data. It is basically an assumption which we generally make about population parameters. We use Hypothesis testing whenever we are trying to draw some inferences on the complete dataset by considering only a sample of the whole population. Null hypothesis and alternate hypothesis are the two main attributes for hypothesis testing. We validate our assumption as statistically effective or not, using the null hypothesis and alternate hypothesis.

There are three types of hypothesis :

- Left tailed
- Right tailed
- Two-tailed test

The type of test depends on the region of the sampling distribution that favours a rejection of H_0. That region is indicated by the alternative hypothesis.

Null hypothesis (H_0)

It is the hypothesis to be tested.

Alternative hypothesis (H_a)

It is a statement of what we assume or predict is true if our sample data cause us to reject the null hypothesis.

Condition for Null hypothesis

Hypothesis	**True**	**False**
Rejection of H_0	Correct	Error type 2
Reject H_0	Error type 1	Correct

Conclusion: - If H_0 is unaccepted or rejected we conclude that H_a is true and accepted. If H_0 is not unaccepted, we conclude that H_0 is true.

Test of Hypothesis concerning difference of means (Two Populations)

Null Hypothesis H_0: $\mu 1 - \mu 2 = g$

Alternative Hypothesis H_a: $\mu 1 - \mu 2 < g$ (Left Tailed) $\mu 1 - \mu 2 > g$ (Right Tailed)
$\mu 1 - \mu 2 \neq g$ (Two Tailed)

$$t = \frac{(\overline{x}_1 - \overline{x}_2) - (\mu_1 - \mu_2)}{s_p \sqrt{\frac{1}{n_1} + \frac{1}{n_2}}}$$

$$s_p^2 = \frac{(n_1 - 1)s_1^2 + (n_2 - 1)s_2^2}{n_1 + n_2 - 2}.$$

Criteria of Rejection: $t < -t_\alpha$ (Left tailed) $t > t_\alpha$ (Right tailed) $|t| > t_{\alpha/2}$ (Two tailed)

METHOD IMPLEMENTATION

<u>Objective 1:</u> <u>To Analyse Employment Generation by Government Policies.</u>

PS1.1: To compare the number of persons registered vs employment provided to them under MGNREGA.

- **Correlation between Registered and Employed Persons.**

```
ps2 = pd.read_csv("ps2_1.csv")
ps2
```

	Year	Total Registered	Total Employed
0	2014	274008304	57685277
1	2015	274960419	67650545
2	2016	256961450	72190136
3	2017	249278655	71565152
4	2018	256807931	73473998
5	2019	273121539	74770328
6	2020	297909825	106431815
7	2021	303833155	92146685

```
data = ps2[['Total Registered','Total Employed']]
correlation = data.corr()
print(correlation)
```

Correlation Coeff. (r) = 0.6857

```
def correlation(r):
    r=float(r)
    if r>-0.1 and r < 0.1:
        print("No Correlation")
    elif r>=0.1 and r<0.5:
        print("Weakly Positive Correlation")
    elif r>=0.5 and r<0.8:
        print("Moderately Positive Correlation")
    elif r>=0.8 and r<1.0:
        print("Strongly Positive Correlation")
    elif r>=-0.1 and r<-0.5:
        print("Weekly Negative Correlation")
    elif r>=-0.5 and r<-0.8:
        print("Moderate Negative Correlation")
    elif r>=-0.8 and r<-1.0:
        print("Moderate Negative Correlation")
```

```
correlation(0.6857)
```

```
Moderately Positive Correlation
```

Conclusion:

The value of r is moderate positive correlation. This concludes that Number of Registered persons are moderately correlated to employed persons.

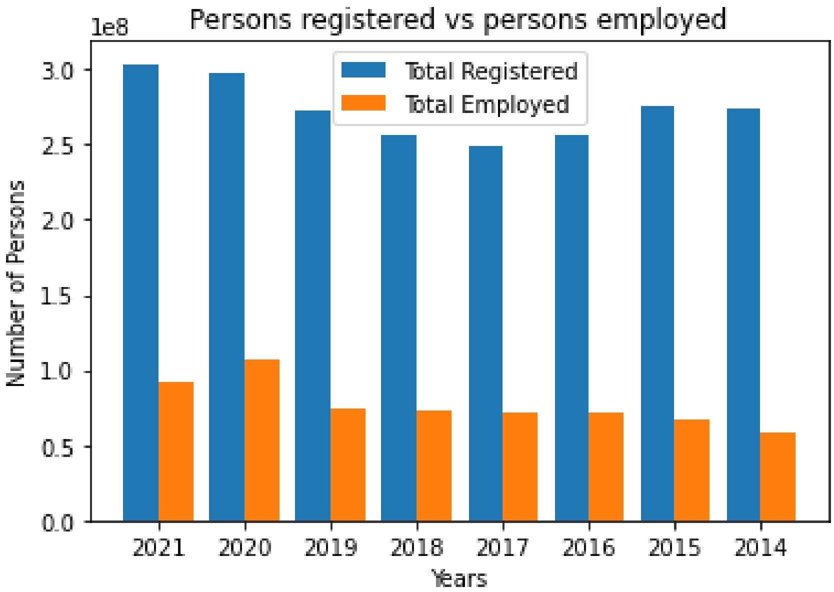


Figure 1: Persons Registered vs Persons Employed

PS1.2: To study employment generated by MGNREGA and predict for future.

- **Prediction using Linear Regression**

Year	Y	X	XY	X^2
2006	21245033	1	21245033	1
2007	33633628	2	67267256	4
2008	44453749	3	133361247	9
2009	52264830	4	209059320	16
2010	54933139	5	274665695	25
2011	78074711	6	468448266	36
2012	74959336	7	524715352	49
2013	69262169	8	554097352	64
2014	57685277	9	519167493	81
2015	67650545	10	676505450	100
2016	72190136	11	794091496	121
2017	71565152	12	858781824	144
2018	73473998	13	955161974	169
2019	74770328	14	1046784592	196
2020	106431815	15	1596477225	225
2021	92146685	16	1474346960	256
Sum	1044740531	136	10174176535	1496
Mean	65296283	8	635886033	94

```
n = len(ps1['X']) - 2
beta = (n * (ps1['XY'].loc['Sum']) - (ps1['X'].loc['Sum'] * ps1['Y'].loc['Sum']))
        / (n * (ps1['X^2'].loc['Sum']) - (ps1['X'].loc['Sum']) ** 2)
print(beta)
```

```
3805535.3573529413
```

```
def reg(x):
    y = ps1['Y'].loc['Mean'] + beta * (x - ps1['X'].loc['Mean'])
    return y
```

```
x_values=[]
y_values=[]

for x in range(23,31):
    x_values.append(x)
    y_values.append(reg(x))

predictions = pd.DataFrame(y_values,[i + 1999 for i in x_values])

predictions.columns=['Predicted Employment']
predictions
```

	Predicted Employment
2022	120476546
2023	124282081
2024	128087617
2025	131893152
2026	135698687
2027	139504223
2028	143309758
2029	147115293

Conclusion :

Figure 2: Persons Employed from 2006 to 2021

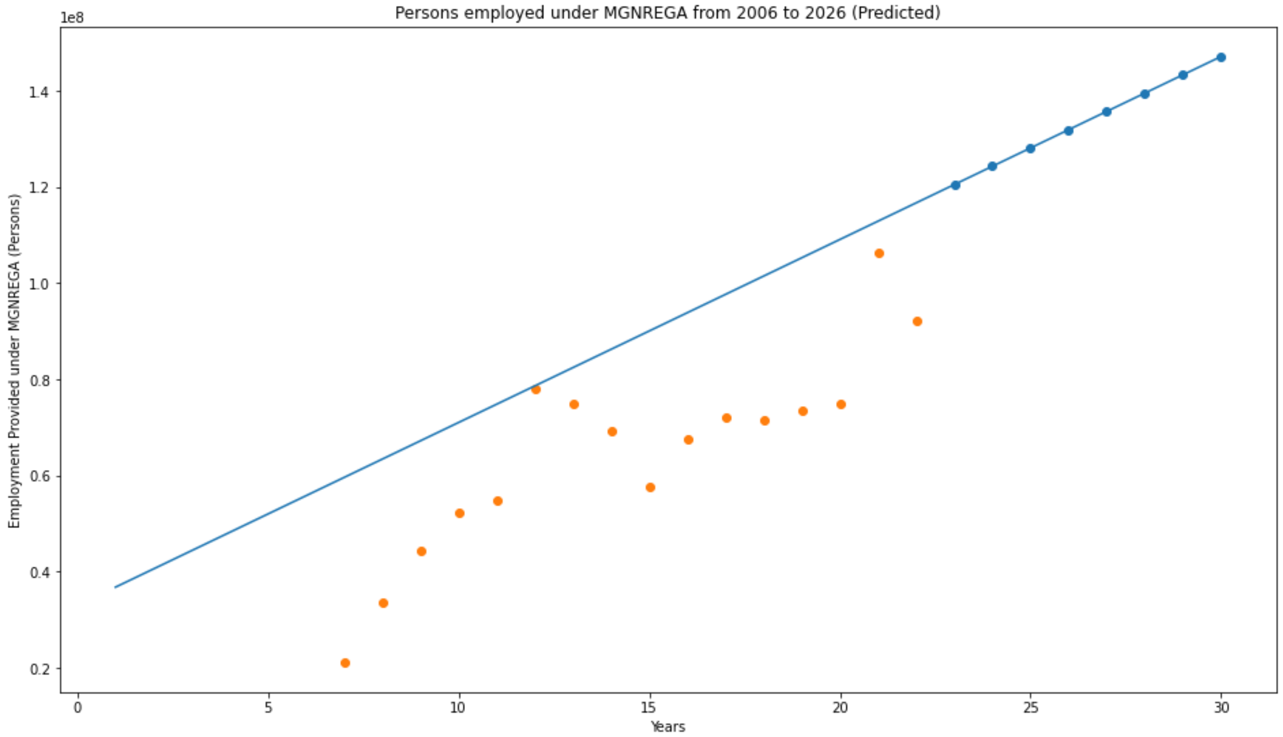


Figure 3: Persons Employed from 2006 to 2029 (Predicted)

PS1.3: To determine the rank of states based on employment generated (per 10 lakh people)

- **Rank by Power Method**

```python
import pandas as pd
mat = pd.read_csv('matrix.csv')
mat
```

	Unnamed: 0	ANDAMAN AND NICOBAR	ANDHRA PRADESH	ARUNACHAL PRADESH	ASSAM	BIHAR	CHHATTISGARH	GOA	GUJARAT	HARYANA	...	ODISHA	PUDUCHERRY	PUNJAB
0	ANDAMAN AND NICOBAR	1	0	0	0	2	0	2	2	2	...	0	2	2
1	ANDHRA PRADESH	2	1	2	2	2	0	2	2	2	...	2	2	2
2	ARUNACHAL PRADESH	2	0	1	2	2	0	2	2	2	...	2	2	2
3	ASSAM	2	0	0	1	2	0	2	2	2	...	0	2	2
4	BIHAR	0	0	0	0	1	0	2	2	2	...	0	0	2
5	CHHATTISGARH	2	2	2	2	2	1	2	2	2	...	2	2	2
6	GOA	0	0	0	0	0	0	1	0	0	...	0	0	0
7	GUJARAT	0	0	0	0	0	0	2	1	2	...	0	0	0
8	HARYANA	0	0	0	0	0	0	2	0	1	...	0	0	0
9	HIMACHAL PRADESH	2	0	0	2	2	0	2	2	2	...	2	2	2
10	JAMMU AND KASHMIR	2	2	2	2	2	2	2	2	2	...	2	2	2
11	JHARKHAND	2	0	0	0	2	0	2	2	2	...	0	2	2
12	KARNATAKA	2	0	0	0	2	0	2	2	2	...	0	2	2
13	KERALA	2	0	0	0	2	0	2	2	2	...	0	2	2
14	LAKSHADWEEP	0	0	0	0	0	0	2	0	0	...	0	0	0
15	MADHYA PRADESH	2	0	0	2	2	0	2	2	2	...	2	2	2
16	MAHARASHTRA	0	0	0	0	0	0	2	2	2	...	0	0	2
17	MANIPUR	2	2	2	2	2	2	2	2	2	...	2	2	2
18	MEGHALAYA	2	2	2	2	2	2	2	2	2	...	2	2	2
19	MIZORAM	2	2	2	2	2	2	2	2	2	...	2	2	2
20	NAGALAND	2	2	2	2	2	2	2	2	2	...	2	2	2
21	ODISHA	2	0	0	2	2	0	2	2	2	...	1	2	2
22	PUDUCHERRY	0	0	0	0	2	0	2	2	2	...	0	1	2
23	PUNJAB	0	0	0	0	0	0	2	2	2	...	0	0	1
24	RAJASTHAN	2	0	0	2	2	0	2	2	2	...	2	2	2
25	SIKKIM	2	0	2	2	2	0	2	2	2	...	2	2	2
26	TAMIL NADU	2	0	0	2	2	0	2	2	2	...	2	2	2
27	TRIPURA	2	2	2	2	2	2	2	2	2	...	2	2	2
28	UTTAR PRADESH	2	0	0	0	2	0	2	2	2	...	0	2	2
29	UTTARAKHAND	2	0	0	0	2	0	2	2	2	...	0	2	2
30	WEST BENGAL	2	0	0	2	2	0	2	2	2	...	2	2	2

31 rows × 32 columns

```
[[1 0 0 0 2 0 2 2 2 0 0 0 0 0 2 0 2 0 0 0 0 0 2 2 0 0 0 0 0 0 0]
 [2 1 2 2 2 0 2 2 2 2 0 2 2 2 2 2 2 0 0 0 0 2 2 2 2 2 2 0 2 2 2]
 [2 0 1 2 2 0 2 2 2 2 0 2 2 2 2 2 2 0 0 0 0 2 2 2 2 0 2 0 2 2 2]
 [2 0 0 1 2 0 2 2 2 0 0 2 2 2 2 0 2 0 0 0 0 0 2 2 0 0 0 0 2 2 0]
 [0 0 0 0 1 0 2 2 2 0 0 0 0 0 2 0 2 0 0 0 0 0 0 2 0 0 0 0 0 0 0]
 [2 2 2 2 2 1 2 2 2 2 0 2 2 2 2 2 2 0 0 0 0 2 2 2 2 2 2 0 2 2 2]
 [0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0]
 [0 0 0 0 0 0 2 1 2 0 0 0 0 0 2 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0]
 [0 0 0 0 0 0 2 0 1 0 0 0 0 0 2 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0]
 [2 0 0 2 2 0 2 2 2 1 0 2 2 2 2 2 2 0 0 0 0 2 2 2 0 0 0 0 2 2 2]
 [2 2 2 2 2 2 2 2 2 2 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2]
 [2 0 0 0 2 0 2 2 2 0 0 1 0 2 2 0 2 0 0 0 0 0 2 2 0 0 0 0 2 2 0]
 [2 0 0 0 2 0 2 2 2 0 0 2 1 2 2 0 2 0 0 0 0 0 2 2 0 0 0 0 2 2 0]
 [2 0 0 0 2 0 2 2 2 0 0 0 0 1 2 0 2 0 0 0 0 0 2 2 0 0 0 0 2 0 0]
 [0 0 0 0 0 0 2 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0]
 [2 0 0 2 2 0 2 2 2 0 0 2 2 2 2 1 2 0 0 0 0 2 2 2 0 0 0 0 2 2 0]
 [0 0 0 0 0 0 2 2 2 0 0 0 0 0 2 0 1 0 0 0 0 0 0 2 0 0 0 0 0 0 0]
 [2 2 2 2 2 2 2 2 2 2 0 2 2 2 2 2 2 1 2 2 0 2 2 2 2 2 2 0 2 2 2]
 [2 2 2 2 2 2 2 2 2 2 0 2 2 2 2 2 2 0 1 2 0 2 2 2 2 2 2 0 2 2 2]
 [2 2 2 2 2 2 2 2 2 2 0 2 2 2 2 2 2 0 0 1 0 2 2 2 2 2 2 0 2 2 2]
 [2 2 2 2 2 2 2 2 2 2 0 2 2 2 2 2 2 2 2 2 1 2 2 2 2 2 2 0 2 2 2]
 [2 0 0 2 2 0 2 2 2 0 0 2 2 2 2 0 2 0 0 0 0 1 2 2 0 0 0 0 2 2 0]
 [0 0 0 0 2 0 2 2 2 0 0 0 0 0 2 0 2 0 0 0 0 0 1 2 0 0 0 0 0 0 0]
 [0 0 0 0 0 0 2 2 2 0 0 0 0 0 2 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0]
 [2 0 0 2 2 0 2 2 2 2 0 2 2 2 2 2 2 0 0 0 0 2 2 2 1 0 0 0 2 2 2]
 [2 0 2 2 2 0 2 2 2 2 0 2 2 2 2 2 2 0 0 0 0 2 2 2 2 1 2 0 2 2 2]
 [2 0 0 2 2 0 2 2 2 2 0 2 2 2 2 2 2 0 0 0 0 2 2 2 2 0 1 0 2 2 2]
 [2 2 2 2 2 2 2 2 2 2 0 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 1 2 2 2]
 [2 0 0 0 2 0 2 2 2 0 0 0 0 0 2 0 2 0 0 0 0 0 2 2 0 0 0 0 1 0 0]
 [2 0 0 0 2 0 2 2 2 0 0 0 0 2 2 0 2 0 0 0 0 0 2 2 0 0 0 0 2 1 0]
 [2 0 0 2 2 0 2 2 2 0 0 2 2 2 2 2 2 0 0 0 0 2 2 2 0 0 0 0 2 2 1]]
```

```
[array([[1],
       [1],
       [1],
       [1],
       [1],
       [1],
       [1],
       [1],
       [1],
       [1],
       [1],
       [1],
       [1],
       [1],
       [1],
       [1],
       [1],
       [1],
       [1],
       [1],
       [1],
       [1],
       [1],
       [1],
       [1],
       [1],
       [1],
       [1],
       [1],
       [1],
       [1]])]
```

```
Iteration 1 :
[[0.2787]
 [0.7705]
 [0.7049]
 [0.4754]
 [0.2131]
 [0.8033]
 [0.0164]
 [0.1148]
 [0.082 ]
 [0.6066]
 [1.    ]
 [0.4098]
 [0.4426]
 [0.3443]
 [0.0492]
 [0.541 ]
 [0.1803]
 [0.9016]
 [0.8689]
 [0.8361]
 [0.9344]
 [0.5082]
 [0.2459]
 [0.1475]
 [0.6393]
 [0.7377]
 [0.6721]
 [0.9672]
 [0.3115]
 [0.377 ]
 [0.5738]]        [61]
```

```
Iteration 2 :
[[7.790e-02]
 [5.938e-01]
 [4.970e-01]
 [2.262e-01]
 [4.570e-02]
 [6.454e-01]
 [5.000e-04]
 [1.340e-02]
 [7.000e-03]
 [3.681e-01]
 [1.000e+00]
 [1.682e-01]
 [1.961e-01]
 [1.188e-01]
 [2.700e-03]
 [2.929e-01]
 [3.280e-02]
 [8.130e-01]
 [7.550e-01]
 [6.991e-01]
 [8.732e-01]
 [2.585e-01]
 [6.070e-02]
 [2.200e-02]
 [4.089e-01]
 [5.443e-01]
 [4.519e-01]
 [9.355e-01]
 [9.730e-02]
 [1.424e-01]
 [3.294e-01]]        [30.5082]
```

```
Iteration 3 :
[[2.200e-02]
 [4.578e-01]
 [3.508e-01]
 [1.079e-01]
 [9.900e-03]
 [5.187e-01]
 [0.000e+00]
 [1.700e-03]
 [7.000e-04]
 [2.237e-01]
 [1.000e+00]
 [6.930e-02]
 [8.720e-02]
 [4.120e-02]
 [2.000e-04]
 [1.588e-01]
 [6.100e-03]
 [7.332e-01]
 [6.562e-01]
 [5.848e-01]
 [8.161e-01]
 [1.318e-01]
 [1.520e-02]
 [3.400e-03]
 [2.619e-01]
 [4.019e-01]
 [3.041e-01]
 [9.049e-01]
 [3.060e-02]
 [5.400e-02]
 [1.894e-01]]        [20.3554]
```

```
Iteration 4 :
[[6.300e-03]
 [3.533e-01]
 [2.479e-01]
 [5.170e-02]
 [2.200e-03]
 [4.172e-01]
 [0.000e+00]
 [2.000e-04]
 [1.000e-04]
 [1.362e-01]
 [1.000e+00]
 [2.870e-02]
 [3.900e-02]
 [1.440e-02]
 [0.000e+00]
 [8.640e-02]
 [1.200e-03]
 [6.615e-01]
 [5.706e-01]
 [4.894e-01]
 [7.628e-01]
 [6.740e-02]
 [3.900e-03]
 [6.000e-04]
 [1.680e-01]
 [2.971e-01]
 [2.050e-01]
 [8.754e-01]
 [9.700e-03]
 [2.070e-02]
 [1.092e-01]]        [15.287]
```

```
Iteration 5 :
[[1.900e-03]
 [2.730e-01]
 [1.755e-01]
 [2.500e-02]
 [5.000e-04]
 [3.359e-01]
 [0.000e+00]
 [0.000e+00]
 [0.000e+00]
 [8.320e-02]
 [1.000e+00]
 [1.200e-02]
 [1.750e-02]
 [5.100e-03]
 [0.000e+00]
 [4.720e-02]
 [2.000e-04]
 [5.970e-01]
 [4.964e-01]
 [4.099e-01]
 [7.132e-01]
 [3.470e-02]
 [1.000e-03]
 [1.000e-04]
 [1.080e-01]
 [2.199e-01]
 [1.385e-01]
 [8.469e-01]
 [3.200e-03]
 [8.000e-03]
 [6.320e-02]]        [12.2522]
```

```
Iteration 6 :
[[5.000e-04]
 [2.113e-01]
 [1.245e-01]
 [1.210e-02]
 [1.000e-04]
 [2.708e-01]
 [0.000e+00]
 [0.000e+00]
 [0.000e+00]
 [5.100e-02]
 [1.000e+00]
 [5.100e-03]
 [8.000e-03]
 [1.800e-03]
 [0.000e+00]
 [2.600e-02]
 [0.000e+00]
 [5.391e-01]
 [4.322e-01]
 [3.437e-01]
 [6.671e-01]
 [1.790e-02]
 [3.000e-04]
 [0.000e+00]
 [6.970e-02]
 [1.631e-01]
 [9.380e-02]
 [8.195e-01]
 [1.000e-03]
 [3.100e-03]
 [3.670e-02]]        [10.234]
```

```
Iteration 7 :
[[1.000e-04]
 [1.638e-01]
 [8.850e-02]
 [5.900e-03]
 [0.000e+00]
 [2.186e-01]
 [0.000e+00]
 [0.000e+00]
 [0.000e+00]
 [3.140e-02]
 [1.000e+00]
 [2.100e-03]
 [3.600e-03]
 [6.000e-04]
 [0.000e+00]
 [1.430e-02]
 [0.000e+00]
 [4.871e-01]
 [3.766e-01]
 [2.884e-01]
 [6.242e-01]
 [9.300e-03]
 [1.000e-04]
 [0.000e+00]
 [4.510e-02]
 [1.212e-01]
 [6.370e-02]
 [7.932e-01]
 [3.000e-04]
 [1.200e-03]
 [2.140e-02]]        [8.7968]
```

```
Iteration 8 :
[[0.000e+00]
 [1.271e-01]
 [6.300e-02]
 [2.800e-03]
 [0.000e+00]
 [1.766e-01]
 [0.000e+00]
 [0.000e+00]
 [0.000e+00]
 [1.930e-02]
 [1.000e+00]
 [9.000e-04]
 [1.600e-03]
 [2.000e-04]
 [0.000e+00]
 [7.900e-03]
 [0.000e+00]
 [4.403e-01]
 [3.284e-01]
 [2.423e-01]
 [5.842e-01]
 [4.800e-03]
 [0.000e+00]
 [0.000e+00]
 [2.920e-02]
 [9.020e-02]
 [4.330e-02]
 [7.678e-01]
 [1.000e-04]
 [4.000e-04]
 [1.250e-02]]        [7.7214]
```

```
Iteration 9 :
[[0.000e+00]
 [9.870e-02]
 [4.490e-02]
 [1.300e-03]
 [0.000e+00]
 [1.428e-01]
 [0.000e+00]
 [0.000e+00]
 [0.000e+00]
 [1.190e-02]
 [1.000e+00]
 [3.000e-04]
 [7.000e-04]
 [1.000e-04]
 [0.000e+00]
 [4.300e-03]
 [0.000e+00]
 [3.981e-01]
 [2.865e-01]
 [2.036e-01]
 [5.469e-01]
 [2.400e-03]
 [0.000e+00]
 [0.000e+00]
 [1.890e-02]
 [6.710e-02]
 [2.940e-02]
 [7.433e-01]
 [0.000e+00]
 [1.000e-04]
 [7.200e-03]]        [6.8858]
```

```
Iteration 10 :
[[0.000e+00]
 [7.650e-02]
 [3.190e-02]
 [6.000e-04]
 [0.000e+00]
 [1.154e-01]
 [0.000e+00]
 [0.000e+00]
 [0.000e+00]
 [7.200e-03]
 [1.000e+00]
 [1.000e-04]
 [3.000e-04]
 [0.000e+00]
 [0.000e+00]
 [2.300e-03]
 [0.000e+00]
 [3.601e-01]
 [2.499e-01]
 [1.711e-01]
 [5.121e-01]
 [1.200e-03]
 [0.000e+00]
 [0.000e+00]
 [1.210e-02]
 [4.990e-02]
 [1.990e-02]
 [7.196e-01]
 [0.000e+00]
 [0.000e+00]
 [4.100e-03]]        [6.217]
```

Conclusion :

	States
1	JAMMU AND KASHMIR
2	TRIPURA
3	NAGALAND
4	MANIPUR
5	MEGHALAYA
6	MIZORAM
7	CHHATTISGARH
8	ANDHRA PRADESH
9	SIKKIM
10	ARUNACHAL PRADESH
11	TAMIL NADU
12	RAJASTHAN
13	HIMACHAL PRADESH
14	WEST BENGAL
15	MADHYA PRADESH
16	ODISHA
17	ASSAM
18	KARNATAKA
19	JHARKHAND
20	UTTARAKHAND
21	KERALA
22	UTTAR PRADESH
23	ANDAMAN AND NICOBAR
24	PUDUCHERRY
25	BIHAR
26	MAHARASHTRA
27	PUNJAB
28	GUJARAT
29	HARYANA
30	LAKSHADWEEP
31	GOA

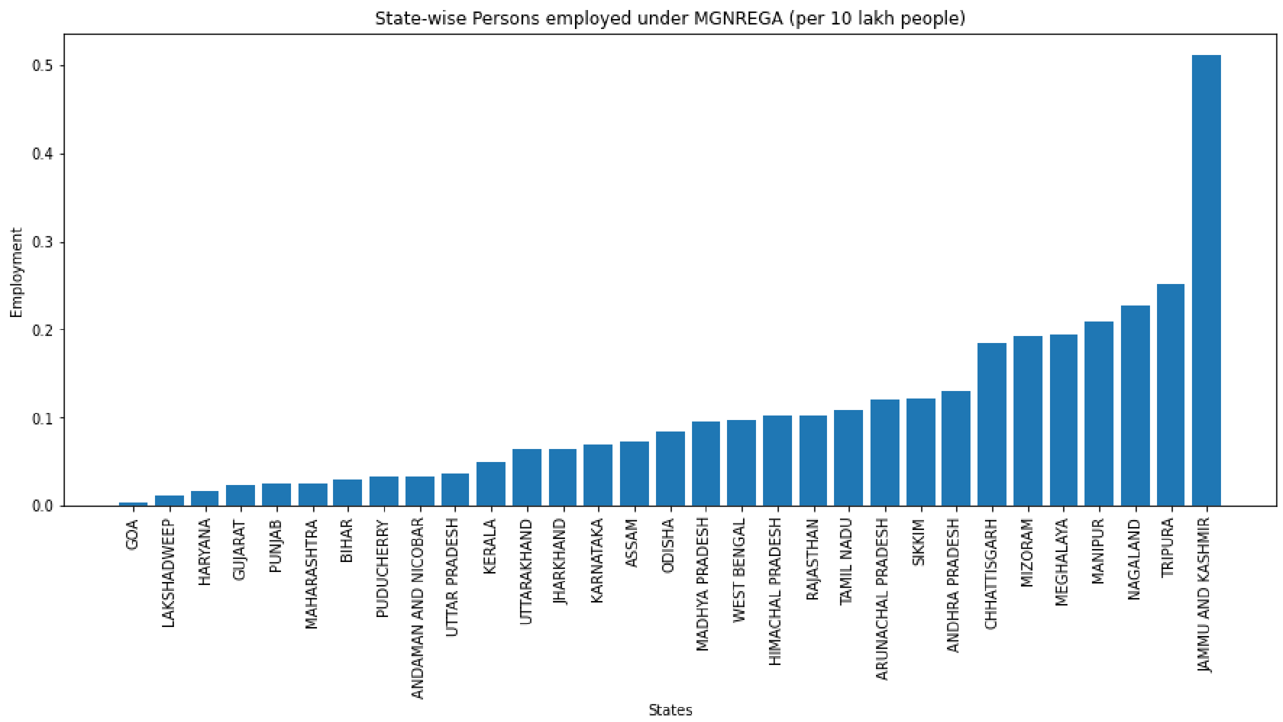


Figure 4: State-wise Ranking under MGNREGA

Objective 2: To analyze how start-ups affect economy, employment, and its different aspects.

PS2.1: To analyze the effect of start-ups on the unemployment rate in India between 2016 and 2020

Claim: To test the hypothesis that there is no correlation between the number of start-ups and the unemployment rate in India.

Step1: Forming hypothesis

- Null Hypothesis: H_o: $\rho = 0$
- Alternate hypothesis: $\rho \neq 0$

Step2: Test statistics

- Here H_o: $\rho = 0$, therefore we going to use two tailed test.
- Level of significance () = 0.05
- Degree of freedom(df) = n - 1 = 4
- Now as our sample size i.e., n < 30 and the population variance is unknown, so we will use t test to find the difference of mean.
- On calculating through t distribution table for the given value of significance and df, the value t_α is found to be 2.132.

Step3: Calculation using python

In [10]:

```
ps1=pd.read_csv("ps1.csv")
ps1
```

Out[10]:

	Calander Year	Recognized Starts-ups	unemployment Rate
0	2016	504	5.662
1	2017	5411	5.549
2	2018	8944	5.370
3	2019	11688	5.371
4	2020	14770	6.469

Calculation of correlation coefficient

In [74]:

```
correlationcoefficient1=np.corrcoef(ps1['Recognized Starts-ups'],ps1['unemployment Rate'])
correlationcoefficient1
print("Corelation coefficient between the Recognized Starts-ups and GDP per capita(in $) is",correlationcoefficient1[0][1])
```

```
Corelation coefficient between the Recognized Starts-ups and GDP per capita(in $) is 0.43311675394123045
```

In [75]:

```
def CR(r):
    r=float(r)
    if r>-0.1 and r<0.1:
        print("No Correlation")
    elif r>=0.1 and r<0.5:
        print("Weakly Positive Correlation")
    elif r>=0.5 and r<0.8:
        print("Moderately Positive Correlation")
    elif r>=0.8 and r<1.0:
        print("Strongly Positive Correlation")
    elif r>=-0.1 and r<-0.5:
        print("Weekly Negative Correlation")
    elif r>=-0.5 and r<-0.8:
        print("Moderate Negative Correlation")
    elif r>=-0.8 and r<-1.0:
        print("Moderate Negative Correlation")
CR(correlationcoefficient1[0][1])
```

```
Weakly Positive Correlation
```

Calculation of deltaR

In [83]:

```
deltar=((correlationcoefficient1[0][1])*((3)**1/2))/((1-(correlationcoefficient1[0][1])**2)**1/2)
deltar
```

Out[83]: 1.5993777253096053

Step4: Criteria of rejection

In [ ]:

```
talpha1==2.132
if deltar>talpha1 or deltar<talpha:
    print("null hypothesis is rejected")
else:
    print("null hypothesis is accepted")
```

Since our delta is greater is smaller than t_α.
Our null hypothesis is rejected

Step5: conclusion
Form the above test of hypothesis, we have seen that our null hypothesis is rejected. And hence we can say that there is a weakly positive correlation between the number of start-ups and the unemployment rate in India between the years 2016 and 2020.

Note: As the size of data collected is small and due the pandemic there is sudden spike in the unemployment rate in India. Therefore, it may be possible that there is correlation or negative correlation between number of start-ups and the unemployment rate.

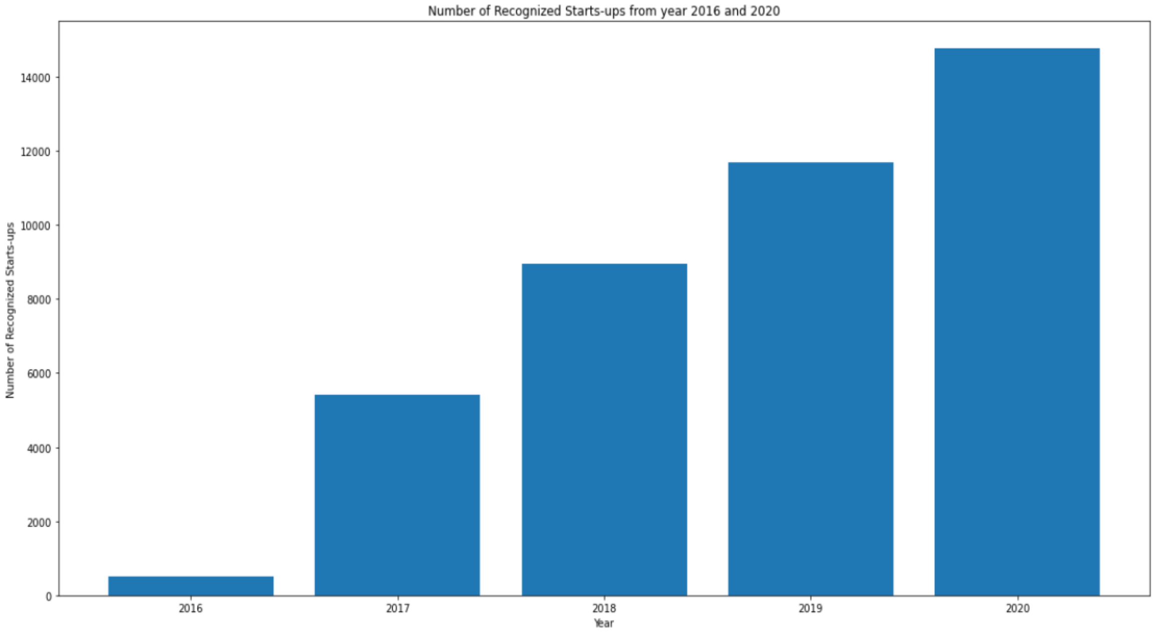


Figure 5: Number of Recognized start-ups

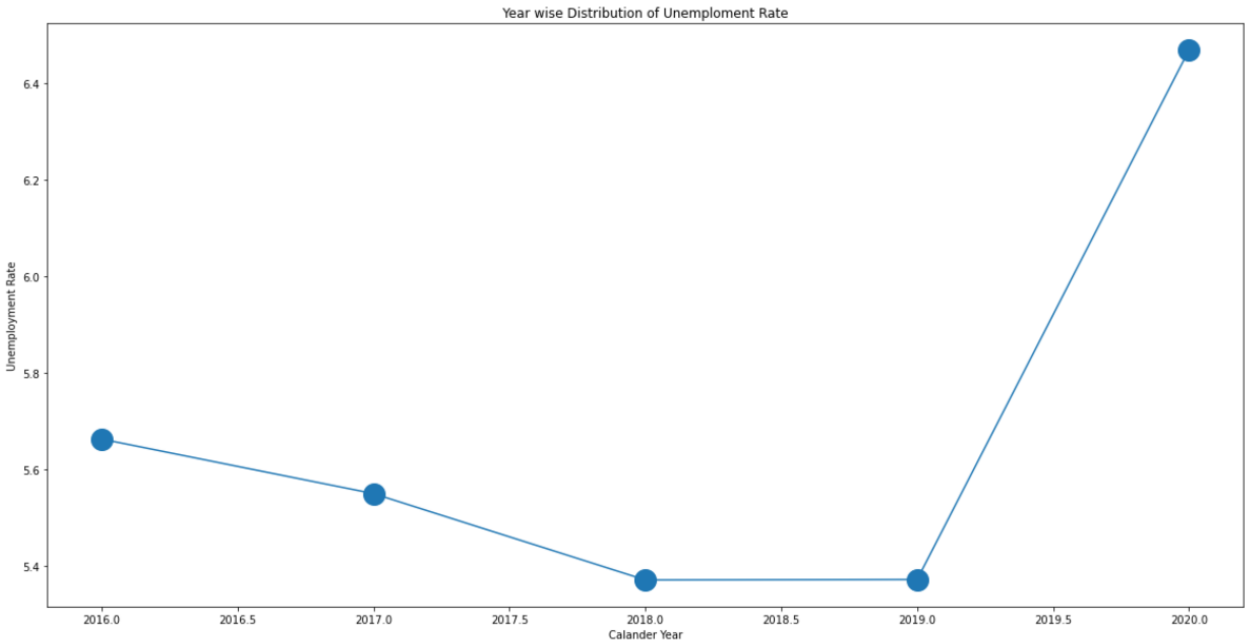


Figure 6: Year Wise Distribution of Start-ups in India

PS2.2: To analyze effect of start-ups on per capita GDP on India between the year 2016 and 2020.

Aim:

To find out the correlation coefficient between number of recognized start-up and GDP per capita using Python programing language.

In [72]:
```
ps2=pd.read_csv("ps2.csv")
ps2
```

Out[72]:

	Calander Year	Recognized Starts-ups	GDP per capita(in $)
0	2016	504	1733
1	2017	5411	1981
2	2018	8944	1997
3	2019	11688	2101
4	2020	14770	1901

In [23]:
```
correlationcoefficient=np.corrcoef(ps2['Recognized Starts-ups'],ps2['GDP per capita(in $)'])
correlationcoefficient
print("Corelation coefficient between the Recognized Starts-ups and GDP per capita(in $) is",correlationcoefficient[0][1])
```

```
Corelation coefficient between the Recognized Starts-ups and GDP per capita(in $) is 0.600646417758553
```

In [33]:
```
def CR(r):
    r=float(r)
    if r>-0.1 and r<0.1:
        print("No Correlation")
    elif r>=0.1 and r<0.5:
        print("Weakly Positive Correlation")
    elif r>=0.5 and r<0.8:
        print("Moderately Positive Correlation")
    elif r>=0.8 and r<1.0:
        print("Strongly Positive Correlation")
    elif r>=-0.1 and r<-0.5:
        print("Weekly Negative Correlation")
    elif r>=-0.5 and r<-0.8:
        print("Moderate Negative Correlation")
    elif r>=-0.8 and r<-1.0:
        print("Moderate Negative Correlation")
```

In [34]:
```
CR(correlationcoefficient[0][1])
```

```
Moderately Positive Correlation
```

Conclusion:

By using python programming language, we have found that the **Correlation coefficient(R)** between the Recognized Starts-ups and GDP per capita (in $) is **0.600646417758553.**

As the correlation coefficient is 0.600646417758553, therefore there is moderately positive correlation

The plot between the Recognized Starts-ups (X) and GDP per capita (in $) (Y)

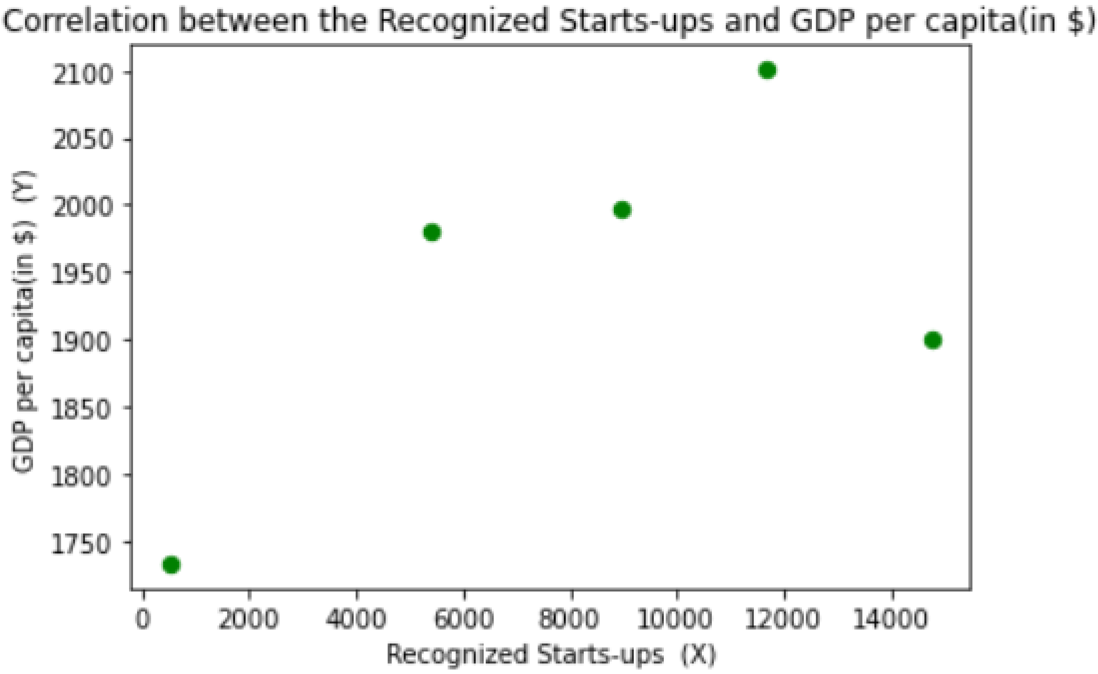


Figure 7: Correlation Between the number of recognized Start-ups and GDP per capita (in $)

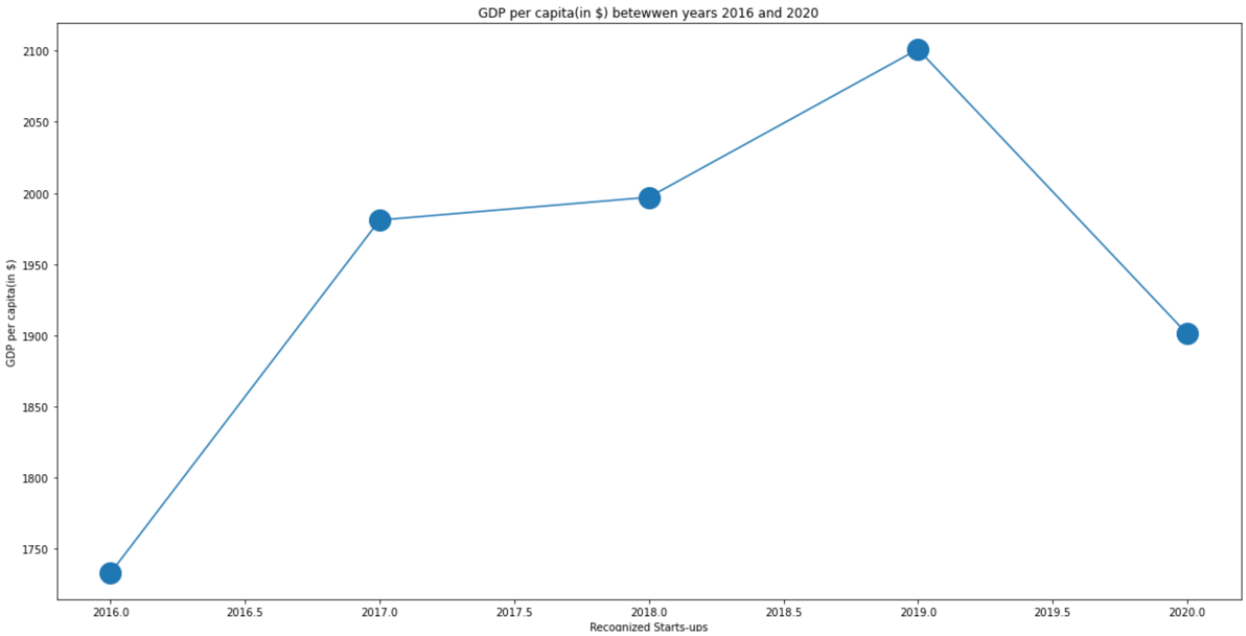


Figure 8: GDP per capita (in $) from year 2016 to 2020

PS2.3: To check how successful investments in start-ups are in India between the years 2016 and 2020.

Claim: To test the hypothesis that the investment in start-ups in India is greater than the revenue generated by start-ups.

Step1: Forming Hypothesis

- Find the mean of Investments in Start-ups and Revenue Generated by Start-ups from 2016 to 2020.
- Let μ_2 and μ_1 be the mean Investments in Start-ups and Revenue Generated by Start-ups respectively.
- Let S1 and S2 be the standard deviation of mean of Investments in Start-ups and Revenue Generated by Start-ups respectively.
- Null hypothesis H_o: $\mu_1 \leq \mu_2$
- Alternate hypothesis H_a: $\mu_1 > \mu_2$

Step2: Test statistics

- Here $\mu_1 - \mu_2 \geq 0$, therefore we going to use left tailed test.
- Level of significance ( ) = 0.05
- Degree of freedom(df) = n - 1 = 4
- Now as our sample size i.e., n < 30 and the population variance is unknown, so we will use t test to find the difference of mean.
- On calculating through t distribution table for the given value of significance and df, the value t_α is found to be 2.132.

Step3: Calculating using python

In [60]:
```
ps3=pd.read_csv("ps3.csv")
ps3
```

Out[60]:

	YEAR	Recognized Starts-ups	Investment in startups(USD amount in billion)	Revenue Genrated by startups (in USD billion)
0	2016	504	4.20	4.5
1	2017	5411	4.30	10.6
2	2018	8944	8.70	10.9
3	2019	11688	8.90	13.0
4	2020	14770	10.14	11.2

In [67]:
```
from scipy import stats
t,a=stats.ttest_rel(ps3['Investment in startups(USD amount in billion)']
                    ,ps3['Revenue Genrated by startups (in USD billion)'])
t
```

Out[67]: -2.5714824295484084

By using python

The value of t= -2.5714824295484084

Step4: Criteria of rejection

In [70]:
```
talpha=2.132
if t>=talpha:
    print("accept null hypothesis")
else:
    print("reject null hypothesis")
```

```
reject null hypothesis
```

Since, the null hypothesis is rejected, through the criteria of left-tailed test, and the alternative hypothesis is accepted.

Step5: Conclusion

From the test of hypothesis, we can conclude that our null hypothesis is rejected, and hence the investment in start-ups is less than the revenue generated by start-ups.

Hence the is investment in start-ups are successful.

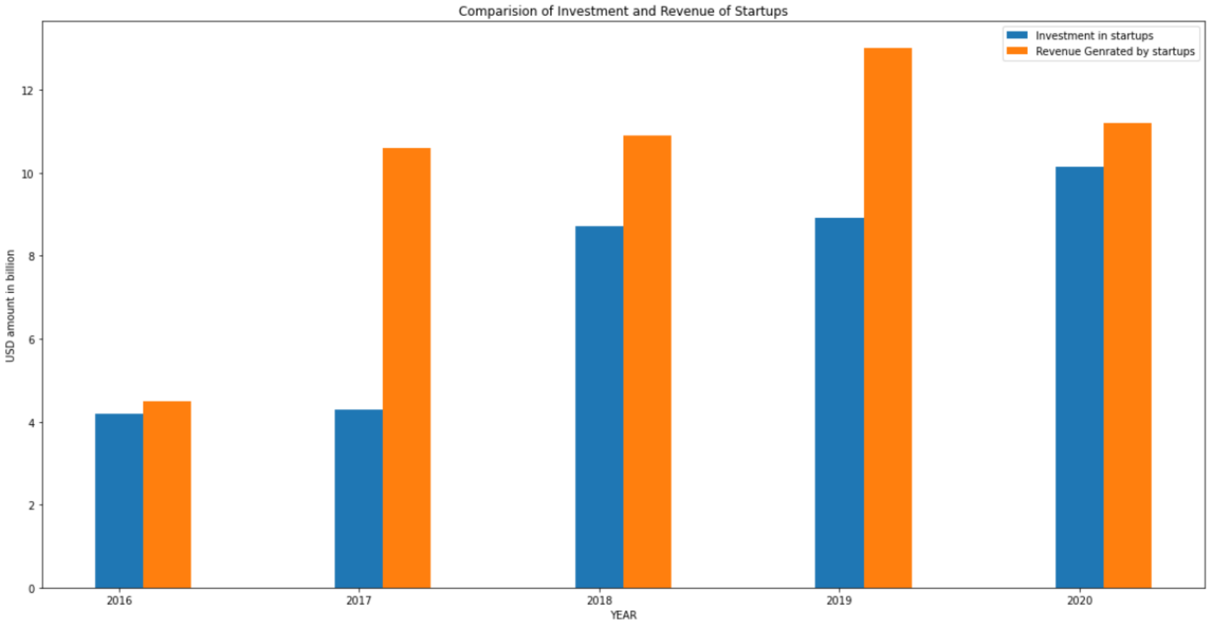


Figure 9: Comparison of investment in start-ups and revenue generated by start-ups from year 2016 to 2020.

Objective 3: To analyze how MSMEs affect Economy and Employment.

PS3.1: To analyze employment generated by MSMEs in India.

- **Rank Correlation State-wise**

```
In [19]: # Import scipy.stats
import scipy.stats

# Two lists x and y
x = [289151.4,3881.8,28309.8,514537,71047.6,16214.6,886022.4,291112.6,31649.8,134822,504166.6,146813.2,585720.6,1627488.2,47007.2
y = [49577103,1383727,31205576,104099452,25545198,1458545,60439692,25351462,6864602,32988134,61095297,33406061,72626809,112374333
print("Spearman's correlation:",scipy.stats.spearmanr(x, y)[0])

Spearman's correlation: 0.8975164499199446
```

Spearman’s correlation:0.8975164499199446

```
In [20]: # Import pandas and scipy.stats
mport pandas as pd
mport scipy.stats

# Two lists x and y
: = [289151.4,3881.8,28309.8,514537,71047.6,16214.6,886022.4,291112.6,31649.8,134822,504166.6,146813.2,585720.6,1627488.2,47007.2,
' = [49577103,1383727,31205576,104099452,25545198,1458545,60439692,25351462,6864602,32988134,61095297,33406061,72626809,112374333,

# Create a function that takes in x's and y's
lef spearmans_rank_correlation(x, y):

    # Calculate the rank of x's
    xranks = pd.Series(x).rank()
    print("Rankings of X:")
    print(xranks)

    # Calculate the ranking of the y's
    yranks = pd.Series(y).rank()
    print("Rankings of Y:")
    print(yranks)

    # Calculate Pearson's correlation coefficient on the ranked versions of the data
    print("Spearman's Rank correlation:",scipy.stats.pearsonr(xranks, yranks)[0])

# Call the function
pearmans_rank_correlation(x, y)
```

```
Rankings of X:
0     25.0
1      6.0
2     16.0
3     32.0
4     20.0
5     14.0
6     35.0
7     26.0
8     17.0
9     21.0
10    31.0
11    23.0
12    33.0
13    37.0
14    18.0
15     5.0
16     7.0
17     4.0
18    22.0
19    24.0
20    30.0
21     3.0
22    36.0
23    29.0
24     9.0
25    34.0
26    19.0
27    27.0
28     8.0
29    15.0
30    11.0
31    12.0
```

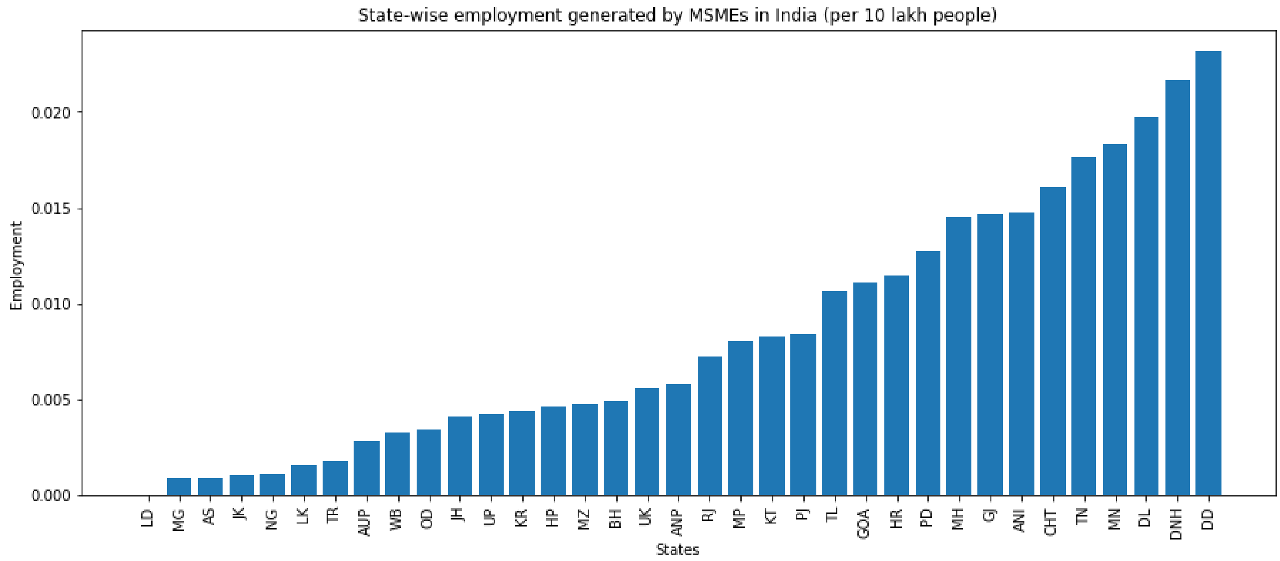
```
32    28.0
33    10.0
34     1.0
35     2.0
36    13.0
dtype: float64
Rankings of Y:
0     28.0
1     10.0
2     23.0
3     35.0
4     21.0
5     11.0
6     29.0
7     20.0
8     16.0
9     24.0
10    30.0
11    25.0
12    33.0
13    36.0
14    13.0
15    14.0
16     8.0
17    12.0
18    27.0
19    22.0
20    31.0
21     6.0
22    32.0
23    26.0
24    15.0
25    37.0
```

```
13    36.0
14    13.0
15    14.0
16     8.0
17    12.0
18    27.0
19    22.0
20    31.0
21     6.0
22    32.0
23    26.0
24    15.0
25    37.0
26    17.0
27    34.0
28     3.0
29     7.0
30     4.5
31     4.5
32    19.0
33    18.0
34     2.0
35     1.0
36     9.0
dtype: float64
Spearman's Rank correlation: 0.8975164499199446
```

CONCLUSION

Out[25]:

	Rank of x	States
0	1.0	LADAKH
1	2.0	LAKSHADWEEP
2	3.0	SIKKIM
3	4.0	NAGALAND
4	5.0	MEGHALAYA
5	6.0	ARUNACHAL PRADESH
6	7.0	MIZORAM
7	8.0	ANDAMAN AND NICOBAR ISLANDS
8	9.0	TRIPURA
9	10.0	JAMMU AND KASHMIR
10	11.0	DADAR AND NAGAR HAVELI
11	12.0	DAMAN AND DIU
12	13.0	PUDUCHERRY
13	14.0	GOA
14	15.0	CHANDIGARH
15	16.0	ASSAM
16	17.0	HIMACHAL PRADESH
17	18.0	MANIPUR
18	19.0	UTTARAKHAND
19	20.0	CHHATTISGARH
20	21.0	JHARKHAND
21	22.0	ODISHA
22	23.0	KERALA
23	24.0	PUNJAB
24	25.0	ANDHRA PRADESH
25	26.0	HARYANA
26	27.0	WEST BENGAL
27	28.0	DELHI
28	29.0	TELANGANA
29	30.0	RAJASTHAN
30	31.0	KARNATAKA
31	32.0	BIHAR
32	33.0	MADHYA PRADESH
33	34.0	UTTAR PRADESH
34	35.0	GUJARAT
35	36.0	TAMIL NADU
36	37.0	MAHARASHTRA
37	NaN	NaN



<u>**Figure 11:** State-wise Employment by MSMEs</u>

PS3.2: To analyze the share of MSMEs in the GDP.

Hypothesis to compare share of MSME in GDP in 2015 and 2019 (paired t-test)

Claim : There is a change in the share of MSME in GDP in the years 2015 and 2019 state-wise.

Out[15]:

	S No	state	2015	2019
0	1	ANP	1.97	2.78
1	2	BH	19.94	2.73
2	3	CHT	0.96	1.20
3	4	GOA	0.16	0.12
4	5	GJ	10.51	7.38
5	6	HR	0.91	2.58
6	7	HP	0.19	0.22
7	8	JH	4.23	1.25
8	9	KT	2.86	4.42
9	10	KR	2.29	1.40
10	11	MP	7.88	12.01
11	12	MH	10.92	27.02
12	13	MN	0.35	0.42
13	14	OD	1.73	1.05
14	15	PJ	0.95	3.41
15	16	RJ	6.84	6.89
16	17	SK	0.01	0.01
17	18	TN	8.41	10.03
18	19	TL	4.14	2.98
19	20	UP	9.10	6.32
20	21	UK	0.36	0.71
21	22	WB	3.31	1.62
22	23	CHT	0.06	0.17
23	24	DNH	0.05	0.06
24	25	DD	0.04	0.04
25	26	DL	1.53	2.33
26	27	PD	0.06	0.13

Step 1: Formation of hypothesis :

First, the mean of the given data has been calculated state wise.
Using python, we can find the mean of the percentage share in 2015 and 2019
Now,
$\mu 1 = Mean$ percentage share in 2015
$\mu 2 = Mean$ percentage share in 2019

$Null\ hypothesis\ H0$: $\mu 2 - \mu 1 = 0$ or $\mu 2 = \mu 1$
$Alternative\ hypothesis\ Ha$: $\mu 2 - \mu 1 > 0$

Step 2: Test statistics :
The corresponding test used in the hypothesis was the paired t-test.

Step 3 : Python Implementation :

In [18]:
```
import pandas as pd
data = pd.read_csv("PS2 GRAPH.csv")
data

data.drop(columns=['state' , 'S No'], inplace = True)
data

data = pd.read_csv("PS2 GRAPH 1.csv")
data[['2015','2019']].describe()
```

```
Out[20]:
```

	2015	2019
count	27.000000	27.000000
mean	3.694815	3.677037
std	4.774684	5.645077
min	0.010000	0.010000
25%	0.270000	0.320000
50%	1.730000	1.620000
75%	5.535000	3.915000
max	19.940000	27.020000

```
In [21]:
from scipy import stats
stats.shapiro(data['2015'])
stats.shapiro(data['2019'])
stats.ttest_rel(data['2015'], data['2019'])

Out[21]: Ttest_relResult(statistic=0.018921395145734484, pvalue=0.9850482706871498)
```

Step 4 : Conclusions :

From the test, we can say that the null hypothesis will be accepted as there is a significant change in the Mean Percentage Share in the years 2015 and 2019. In simple words, we can say that there is a visible change in the share of MSME in GDP between the year 2015 and 2019.

```
23]:
import pandas as pd
from scipy import stats

data[['2015','2019']].describe()
ttest,pval = stats.ttest_rel(data['2015'], data['2019'])
print(pval)

if pval<0.05:
    print("reject null hypothesis")
else:
    print("accept null hypothesis")

0.9850482706871498
accept null hypothesis
```

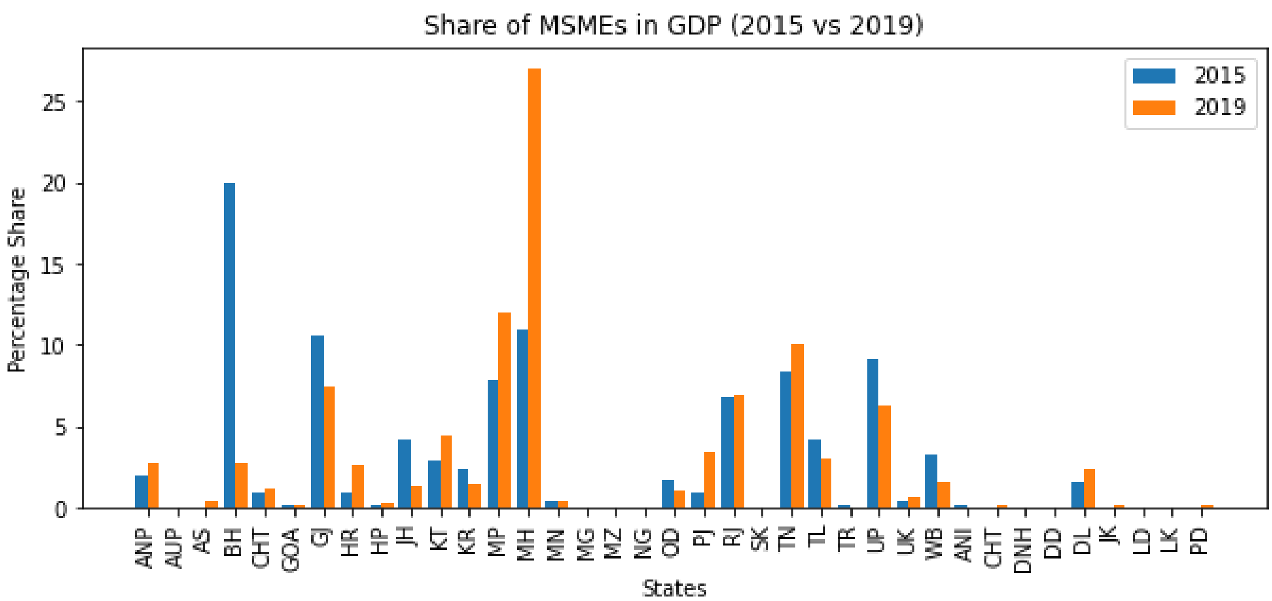


Figure 12: Share of MSMEs in GDP

PS3.3: To compare number of registrations of manufacturing enterprises vs services enterprises.

- **Regression between Manufacturing and Service Enterprises**

```
In [3]: data = pd.read_csv("ps 3 graph.csv")
        data
```

	state	Manufacturing	Services
0	ANP	22271.0	40964.4
1	AUP	172.4	107.8
2	AS	1499.2	1250.0
3	BH	59633.4	115014.8
4	CHT	3680.4	7731.8
5	GOA	662.4	819.4
6	GJ	84364.2	65497.0
7	HR	14633.4	16755.2
8	HP	1212.0	1272.2
9	JH	9491.6	18657.0
10	KT	27308.2	29823.6
11	KR	12618.4	9173.4
12	MP	59282.6	111798.6
13	MH	100806.2	204413.4
14	MN	3593.0	2660.8
15	MG	272.8	127.2
16	MZ	357.2	202.6
17	NG	144.6	95.8
18	OD	7906.2	12378.2
19	PJ	15259.4	20341.6
20	RJ	42830.8	59804.8
21	SK	49.8	103.2
22	TN	87899.0	101479.0
23	TL	18680.6	35132.4
24	TR	533.6	499.2
25	UP	75924.6	85312.4
26	UK	3245.6	3710.2
27	WB	19928.8	19251.8
28	ANI	361.4	776.4
29	CHT	666.4	1267.0
30	DNH	442.4	414.2
31	DD	315.4	152.4
32	DL	12524.0	15067.8
33	JK	515.4	1072.6
34	LD	9.8	8.4
35	LK	824.6	1032.0

	state	X	Y	XY	X^2
0	ANP	22271.0	40964.4	9.123182e+08	4.959974e+08
1	AUP	172.4	107.8	1.858472e+04	2.972176e+04
2	AS	1499.2	1250.0	1.874000e+06	2.247601e+06
3	BH	59633.4	115014.8	6.858724e+09	3.556142e+09
4	CHT	3680.4	7731.8	2.845612e+07	1.354534e+07
5	GOA	662.4	819.4	5.427706e+05	4.387738e+05
6	GJ	84364.2	65497.0	5.525602e+09	7.117318e+09
7	HR	14633.4	16755.2	2.451855e+08	2.141364e+08
8	HP	1212.0	1272.2	1.541906e+06	1.468944e+06
9	JH	9491.6	18657.0	1.770848e+08	9.009047e+07
10	KT	27308.2	29823.6	8.144288e+08	7.457378e+08
11	KR	12618.4	9173.4	1.157536e+08	1.592240e+08
12	MP	59282.6	111798.6	6.627712e+09	3.514427e+09
13	MH	100806.2	204413.4	2.060614e+10	1.016189e+10
14	MN	3593.0	2660.8	9.560254e+06	1.290965e+07
15	MG	272.8	127.2	3.470016e+04	7.441984e+04
16	MZ	357.2	202.6	7.236872e+04	1.275918e+05
17	NG	144.6	95.8	1.385268e+04	2.090916e+04
18	OD	7906.2	12378.2	9.786452e+07	6.250800e+07
19	PJ	15259.4	20341.6	3.104006e+08	2.328493e+08
20	RJ	42830.8	59804.8	2.561487e+09	1.834477e+09
21	SK	49.8	103.2	5.139360e+03	2.480040e+03
22	TN	87899.0	101479.0	8.919903e+09	7.726234e+09
23	TL	18680.6	35132.4	6.562943e+08	3.489648e+08
24	TR	533.6	499.2	2.663731e+05	2.847290e+05
25	UP	75924.6	85312.4	6.477310e+09	5.764545e+09
26	UK	3245.6	3710.2	1.204183e+07	1.053392e+07
27	WB	19928.8	19251.8	3.836653e+08	3.971571e+08
28	ANI	361.4	776.4	2.805910e+05	1.306100e+05
29	CHT	666.4	1267.0	8.443288e+05	4.440890e+05
30	DNH	442.4	414.2	1.832421e+05	1.957178e+05
31	DD	315.4	152.4	4.806696e+04	9.947716e+04
32	DL	12524.0	15067.8	1.887091e+08	1.568506e+08
33	JK	515.4	1072.6	5.528180e+05	2.656372e+05
34	LD	9.8	8.4	8.232000e+01	9.604000e+01
35	LK	824.6	1032.0	8.509872e+05	6.799652e+05
36	PD	689920.8	984168.6	6.789984e+08	4.759907e+07

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns

data = pd.read_csv("ps 3 graph.csv")
data

data.drop(columns=['state'], inplace = True)
data

x = data[['Manufacturing']]
y = data['Services']

from sklearn.model_selection import train_test_split
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size = 0.3, random_state = 100)
2
from sklearn.linear_model import LinearRegression
slr = LinearRegression()
slr.fit(x_train, y_train)

print("Intercept: ", slr.intercept_)
print("Coefficient: ", slr.coef_)

#Prediction of test set
y_pred_slr= slr.predict(x_test)
#Predicted values
print("Prediction for test set: {}".format(y_pred_slr))
```

```
slr_diff = pd.DataFrame({'Actual value': y_test, 'Predicted value': y_pred_slr})
slr_diff


from sklearn import metrics
meanAbErr = metrics.mean_absolute_error(y_test, y_pred_slr)
meanSqErr = metrics.mean_squared_error(y_test, y_pred_slr)
rootMeanSqErr = np.sqrt(metrics.mean_squared_error(y_test, y_pred_slr))
print('R squared: {:.2f}'.format(slr.score(x,y)*100))
print('Mean Absolute Error:', meanAbErr)
print('Mean Square Error:', meanSqErr)
print('Root Mean Square Error:', rootMeanSqErr)
```

```
Intercept:  -1413.723208261501
Coefficient:  [1.48529713]
Prediction for test set: [ 3.16653293e+04 -4.23921198e+02 -7.56627756e+02 -1.88947192e+02
  1.29142410e+05  1.02332366e+06  2.03212239e+04  8.66385527e+04
 -1.39916730e+03 -4.29862387e+02 -1.15765798e+03  2.12510199e+04]
R squared: 98.81
Mean Absolute Error: 9479.83491197433
Mean Square Error: 253524297.08379373
Root Mean Square Error: 15922.44632849468
```

Regression Equation: Services = -1413.723 + 1.48*Manufactoring

Figure 13: <u>Number of Registrations in Manufacturing vs Service Enterprises</u>

<u>Conclusion:</u>

<u>Predicted Registration</u>

```
# regression equation: services = -1413.723 + 1.48*manufacturing
```

```
plt.scatter(x_test,y_test)
plt.plot(x_test, y_pred_slr, 'Red')
plt.show()
```

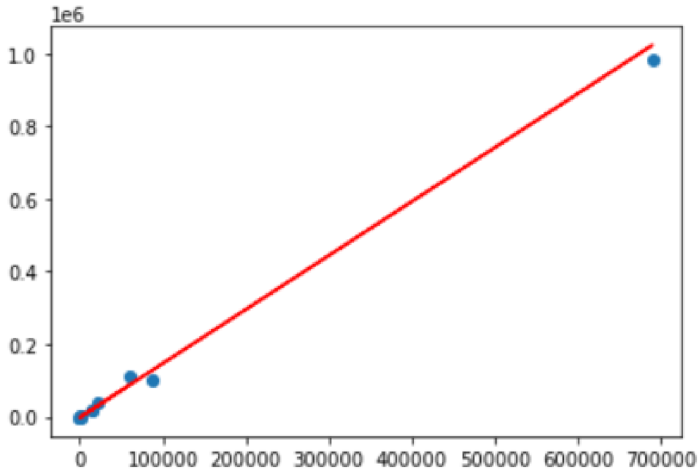


Figure 14: Predicted Regression between Manufacturing vs Service Enterprises

Objective 4: To study employment opportunities created by the tourism industries.

PS4.1: To analyse how Tourism is affecting employment in India.

- **Correlation between Foreign Tourist Arrivals and Employment due to Tourism**

	Years	Total Employees in the tourism sector (in Millions)	Foreign Tourist Arrivals (in Millions)
0	2014	67.20	7.8
1	2015	69.75	8.1
2	2016	72.26	8.8
3	2017	75.34	10.2
4	2018	80.63	10.6
5	2019	87.50	10.9

To find if there is any relation in the two factors Total employment in the tourism Industry and the Foreign Tourist Arrivals.

```
data = dt[['Total Employees in the tourism sector (in Millions)'
           ,'Foreign Tourist Arrivals (in Millions)']]
correlation = data.corr()
print(correlation)
```

From the above coding we can conclude that the correlation coefficient for this is :
r = 0.929348

```python
def correlation(r):
    r=float(r)
    if r>-0.1 and r < 0.1:
        print("No Correlation")
    elif r>=0.1 and r<0.5:
        print("Weakly Positive Correlation")
    elif r>=0.5 and r<0.8:
        print("Moderately Positive Correlation")
    elif r>=0.8 and r<1.0:
        print("Strongly Positive Correlation")
    elif r>=-0.1 and r<-0.5:
        print("Weekly Negative Correlation")
    elif r>=-0.5 and r<-0.8:
        print("Moderate Negative Correlation")
    elif r>=-0.8 and r<-1.0:
        print("Moderate Negative Correlation")
```

```python
correlation(0.929348)
```

```
Strongly Positive Correlation
```

Therefore, we can conclude that these two factors namely “Total employment in the tourism Industry” and “Foreign Tourist Arrivals” are in Strongly Positive Correlation.

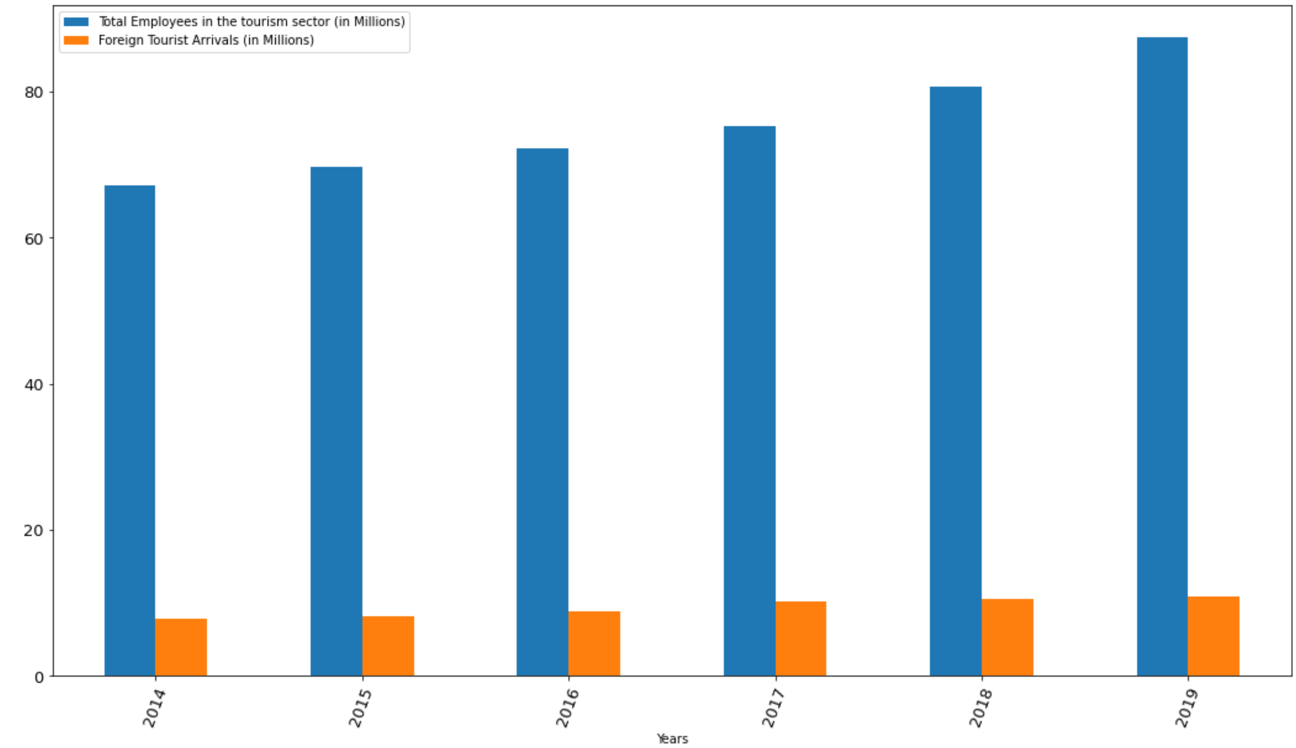


Figure 14 : Total employment in the tourism Industry vs Foreign Tourist Arrivals

PS4.2: To study the contribution of the tourism sector in the GDP.

- **Correlation between Tourism Sector and total share in GDP**

	Years	Total Contribution to GDP (in Billion USD)	Capital Investments in tourism Sector (in Billion USD)
0	2014	120.6	35.0
1	2015	129.5	34.6
2	2016	219.7	36.6
3	2017	232.0	47.8
4	2018	247.4	48.4
5	2019	191.3	42.3
6	2020	121.9	49.0

To find if Total Contribution to GDP and the Capital Investments in tourism Sector has any kind of relation or not.

```
data2 = dt2[['Total Contribution to GDP (in Billion USD)'
            ,'Capital Investments in tourism Sector (in Billion USD)']]
correlation2 = data2.corr()
print(correlation2)
```

From the above coding we can conclude that the correlation coefficient for this is :
r = 0.416396

```
def correlation(r):
    r=float(r)
    if r>-0.1 and r < 0.1:
        print("No Correlation")
    elif r>=0.1 and r<0.5:
        print("Weakly Positive Correlation")
    elif r>=0.5 and r<0.8:
        print("Moderately Positive Correlation")
    elif r>=0.8 and r<1.0:
        print("Strongly Positive Correlation")
    elif r>=-0.1 and r<-0.5:
        print("Weekly Negative Correlation")
    elif r>=-0.5 and r<-0.8:
        print("Moderate Negative Correlation")
    elif r>=-0.8 and r<-1.0:
        print("Moderate Negative Correlation")
```

```
correlation(0.416396)
```

```
Weakly Positive Correlation
```

This implies that the Total Contribution to GDP and the Capital Investments in tourism Sector are Weakly Positive Correlated.

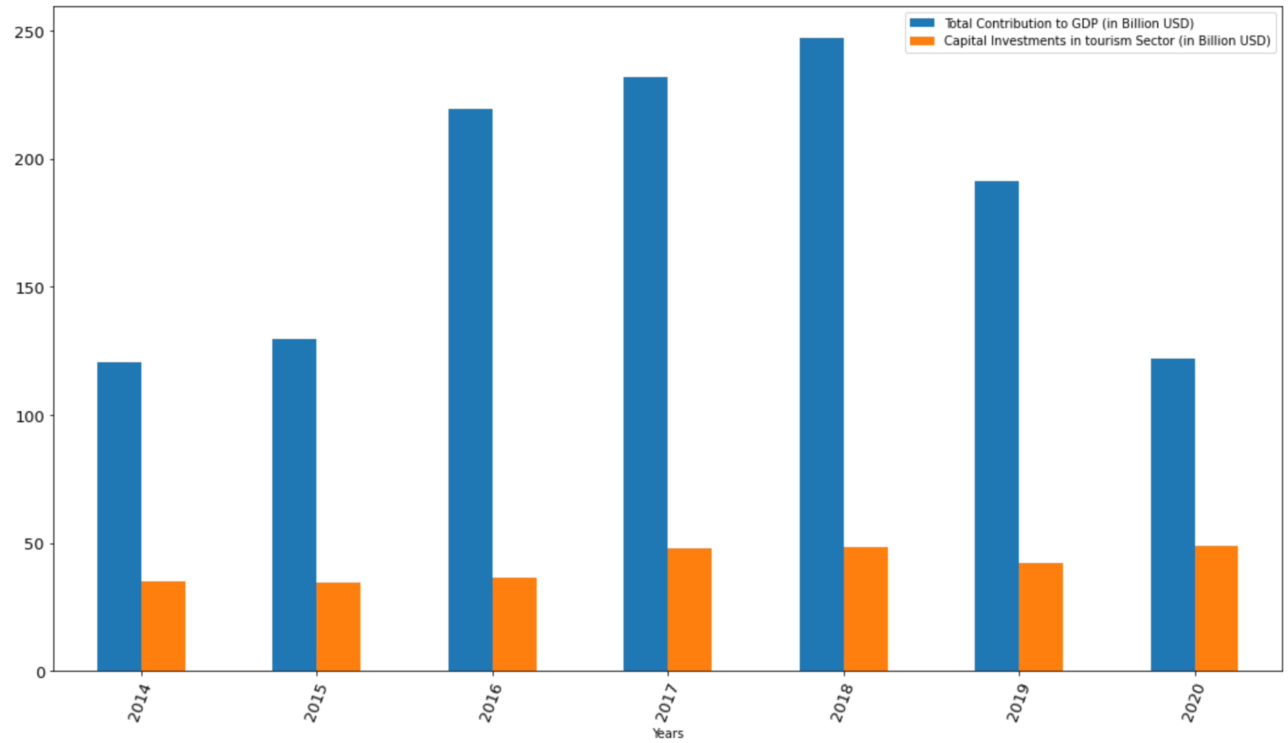


Figure 15: Total Contribution to GDP and Capital Investments in tourism Sector

PS4.3: To compare direct and indirect employment due to Tourism.

- **Hypothesis that Mean Percent is same of Direct and Indirect Employment from 2013 to 2019 (t-test)**

Claim : There is no change in the Mean of the percentage share in the Direct and Indirect in the years 2013 and 2019.

	Years	Shares in Jobs (%)	Indirect (%)	Direct (%)	Direct+Indirect (in Millions)
0	2013-14	11.91	5.19	6.72	67.19
1	2014-15	12.14	5.30	6.84	69.56
2	2015-16	12.38	5.40	6.98	72.26
3	2016-17	12.20	5.32	6.88	75.71
4	2017-18	12.29	5.36	6.93	80.54
5	2018-19	13.00	6.00	7.00	89.00

Step 1: Formation of hypothesis :
First, the mean of the given data has been calculated.
Using python, we can find the mean of the percentage share in Direct and Indirect employment.
Now,
$\mu 1 = Mean$ percentage share in 2013
$\mu 2 = Mean$ percentage share in 2019

*Null hypothesis H*0 : μ2−μ1=0 or $\mu 2= \mu 1$
Alternative hypothesis Ha : $\mu 2-\mu 1>0$

Step 2: Test statistics :
The corresponding test used in the hypothesis was the Two-Tailed Test.

Step 3 : Python Implementation :

```
dt1.describe()
```

	Shares in Jobs (%)	Indirect (%)	Direct (%)	Direct+Indirect (in Millions)
count	6.00000	6.000000	6.000000	6.000000
mean	12.32000	5.428333	6.891667	75.710000
std	0.36927	0.288889	0.103231	8.028305
min	11.91000	5.190000	6.720000	67.190000
25%	12.15500	5.305000	6.850000	70.235000
50%	12.24500	5.340000	6.905000	73.985000
75%	12.35750	5.390000	6.967500	79.332500
max	13.00000	6.000000	7.000000	89.000000

```
import pandas as pd
from scipy import stats

dt1[['Direct (%)','Indirect (%)']].describe()
ttest,pval = stats.ttest_rel(dt1['Direct (%)'], dt1['Indirect (%)'])
print(pval)

if pval<0.05:
    print("reject null hypothesis")
else:
    print("accept null hypothesis")
```

```
1.8828987995349395e-05
reject null hypothesis
```

Step 4 : Conclusions :

From the test, we can say that the null hypothesis will be rejected as there is a significant change in the Mean Percentage Share in the years 2013 and 2019. In simple words, we can say that there is a visible change in the share in the average share percentage in Direct and Indirect Employment.

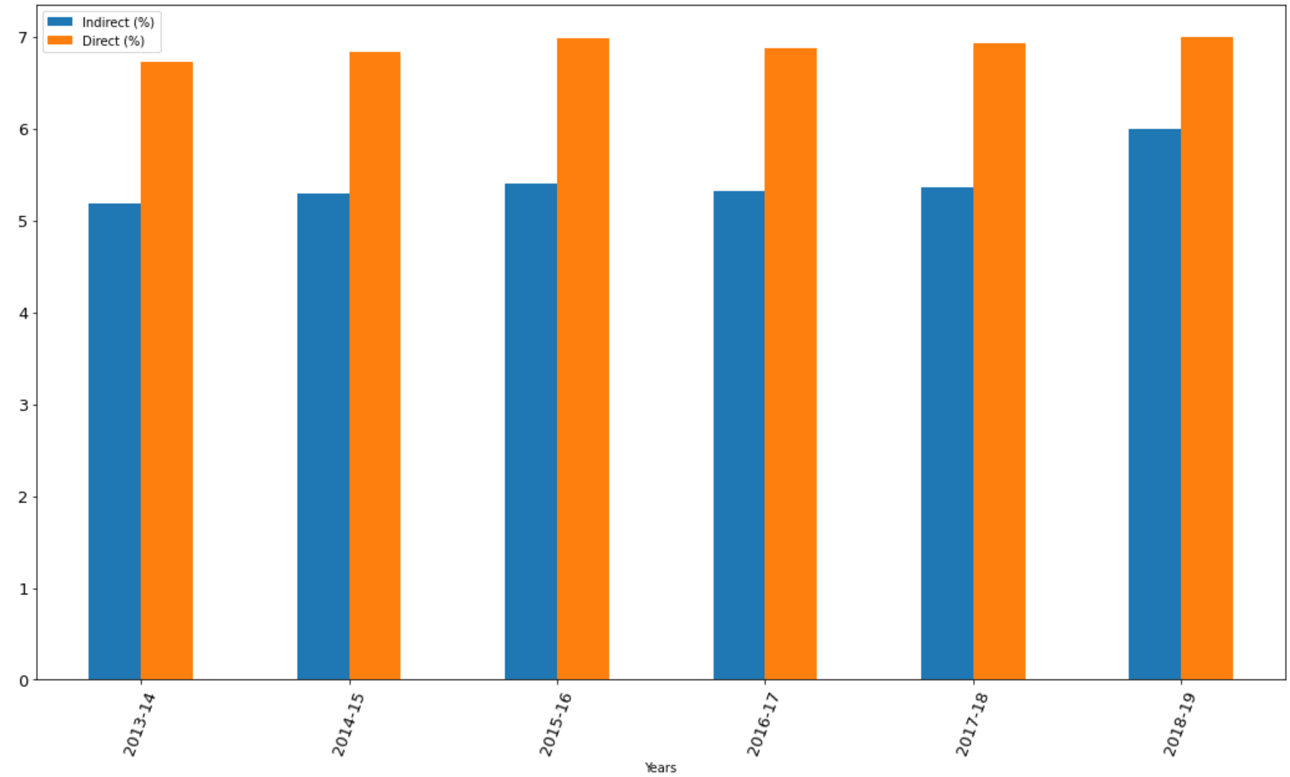


Figure 16: percentage share in Direct and Indirect employment

CONCLUSIONS

On analysing the correlation between number of registries and actual employment under MGNREGA, it resulted to be moderately correlated. This means that the number of employed persons cannot accurately said based on persons registered. Also, on analysing the data on MGNREGA from 2006 to 2021 and then predicting for 2022 to 2029, it was found that there might not be any significant increase in employment after 2022 as the difference between the years is small. On finding the ranking of states based on employment provided under MGNREGA, taking into consideration the population of the state, it concluded Jammu & Kashmir tops in providing employment and many states like Haryana, Goa, Gujarat provides least employment. In the end, we can conclude that MGNREGA which was implemented in 2006 proved to be a success over the years and will continue to do so. The only drawback which we can assume is that there is still a major difference in persons who got registered vs which got employed.

After the above analysis of effect of Start-ups on employment and economy in India, we can conclude that there is a weakly positive correlation between the number of start-ups and unemployment rate between the years 2016 and 2020. And there is a moderately positive correlation between the number of start-ups and GDP per capita between the years 2016 and 2020. Also, the investment in start-ups is successful, as the revenue generated by start-ups is greater than the investments in start-ups from year 2016 to 2020. Hence, we can conclude objective two that start-ups can be a good factor to generate employment and boost Indian economy.

Through the analysis of MSME on employment and the economy of India, we have come to the conclusion that every state contributes to employment and to the GDP at its own pace. Also, the hypothesis claimed every state has shown a significant increase in contribution in GDP India (among years 2015 and 2019). Spearmen correlation rank has shown Daman and Diu has the highest employment whereas Ladakh has the lowest employment. over through regression, we have seen there is an established relationship between factoring and services enterprises.

After analyzing the employment opportunities created by the tourism industries. We can see that the total employment generated is very strongly related to the foreign tourist arrivals as there is an increase with this simultaneously, the more the tourist arrives the more the job opportunity will be there. Similarly, the contribution in GDP doesn't relate much to foreign investments or Capital Investments and is not likely to form any job opportunities in India now. After this, if we compare direct and indirect employment, we can say that the hypothesis was found to be unsatisfactory that there is change in the Mean of the percentage share in the Direct and Indirect in the years 2013 and 2019.

REFERENCES

- MGNREGA Public Data Portal hhttps://nregarep2.nic.in/netnrega/dynamic2/DynamicReport_new4.aspx retrieved on 25 January 2022.
- Study on MGNREGA by V M Kadrolkar (2012) https://www.researchgate.net/publication/235949503 retrieved on 1 February 2022
- Study on MGNREGA by R S Ghuman (2008) https://www.levyinstitute.org/pubs/EFFE/ranjit_singh_ghuman.pdf retrieved on 27 February 2022Study on MGNREGA by R S Ghuman (2008) https://www.levyinstitute.org/pubs/EFFE/ranjit_singh_ghuman.pdf retrieved on 27 January 2022
- Department for Promotion of Industry and Internal Trade (https://dpiit.gov.in/sites/default/files/lu2614.pdf) retrieved on 2 February 2022
- World bank data (https://data.worldbank.org/indicator/SL.UEM.TOTL.ZS)
- Live mint (https://www.livemint.com/news/india/indian-startups-attract-10-14-bn-in-funding-in-2020-report-11611670698780.html) and ADB organisation (https://www.adb.org/sites/default/files/publication/612516/adbi-wp1145.pdf) retrieved on 4 February 2022.
- Macro trends (https://www.macrotrends.net/countries/IND/india/gdp-per-capita)retrieved on 6 February 2022.
- Money control (https://www.moneycontrol.com/news/business/startup/indian-startups-race-past-20-billion-funding-in-record-year-7394701.html) retried on 7 February 2022.
- MSME report by http://www.dcmsme.gov.in/uam_publication_2015-2018.pdf
- For MSME data https://msme.gov.in/
- Statista.com (https://www.statista.com/statistics/799836/india-capital-investments-value-in-the-tourism-sector/) retrieved on 14th February
- India brand equity foundation (https://www.ibef.org/industry/tourism-hospitality-india/infographic) retrieved on 14th February
- Ministry of Tourism Portal(https://tourism.gov.in/market-research-and-statistics) retrieved on 14th February.

APPENDICES

Appendix 1: Objective 1.1

```
ps1 = pd.read_csv("ps1.csv")
ps1
```

	Year	Total Registered	Total Employed
0	2014	274008304	57685277
1	2015	274960419	67650545
2	2016	256961450	72190136
3	2017	249278655	71565152
4	2018	256807931	73473998
5	2019	273121539	74770328
6	2020	297909825	106431815
7	2021	303833155	92146685

```
data = ps1[['Total Registered','Total Employed']]
correlation = data.corr()
print(correlation)
```

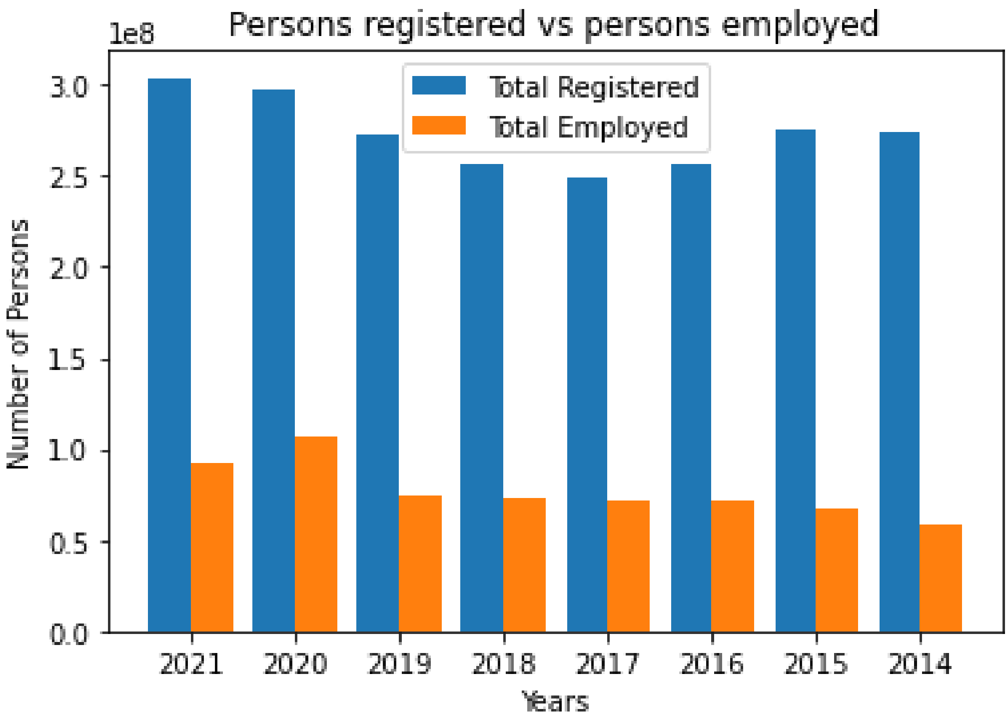
```
                  Total Registered  Total Employed
Total Registered            1.0000          0.6857
Total Employed              0.6857          1.0000
```

```
def correlation(r):
    r=float(r)
    if r>-0.1 and r < 0.1:
        print("No Correlation")
    elif r>=0.1 and r<0.5:
        print("Weakly Positive Correlation")
    elif r>=0.5 and r<0.8:
        print("Moderately Positive Correlation")
    elif r>=0.8 and r<1.0:
        print("Strongly Positive Correlation")
    elif r>=-0.1 and r<-0.5:
        print("Weekly Negative Correlation")
    elif r>=-0.5 and r<-0.8:
        print("Moderate Negative Correlation")
    elif r>=-0.8 and r<-1.0:
        print("Moderate Negative Correlation")
```

```
correlation(0.6857)
```

```
Moderately Positive Correlation
```

```
import numpy as np
import matplotlib.pyplot as plt
years = ['2021', '2020', '2019','2018', '2017', '2016', '2015', '2014']
registered = [303833155, 297909825, 273121539, 256807931, 249278655, 256961450, 274960419, 274008304]
employed = [92146685, 106431815, 74770328, 73473998, 71565152, 72190136, 67650545, 57685277]
X_axis = np.arange(len(years))
plt.bar(X_axis - 0.2, registered, 0.4, label = 'Total Registered')
plt.bar(X_axis + 0.2, employed, 0.4, label = 'Total Employed')
plt.xticks(X_axis, years)
plt.xlabel("Years")
plt.ylabel("Number of Persons")
plt.title("Persons registered vs persons employed")
plt.legend()
plt.show()
```



Appendix 1: Objective 1.2

```
ps1 = pd.read_csv("ps1.csv")
ps1
```

	Year	Total
0	2006	21245033
1	2007	33633628
2	2008	44453749
3	2009	52264830
4	2010	54933139
5	2011	78074711
6	2012	74959336
7	2013	69262169
8	2014	57685277
9	2015	67650545
10	2016	72190136
11	2017	71565152
12	2018	73473998
13	2019	74770328
14	2020	106431815
15	2021	92146685

```
ps1 = ps1.set_index('Year')
```

```
ps1['Y'] = ps1['Total']
ps1['X'] = range(1,len(ps1['Total'])+1)
ps1['XY'] = ps1['Y'] * ps1['X']
ps1['X^2'] = ps1['X'] * ps1['X']
ps1
```

Year	Total	Y	X	XY	X^2
2006	21245033	21245033	1	21245033	1
2007	33633628	33633628	2	67267256	4
2008	44453749	44453749	3	133361247	9
2009	52264830	52264830	4	209059320	16
2010	54933139	54933139	5	274665695	25
2011	78074711	78074711	6	468448266	36
2012	74959336	74959336	7	524715352	49
2013	69262169	69262169	8	554097352	64
2014	57685277	57685277	9	519167493	81
2015	67650545	67650545	10	676505450	100
2016	72190136	72190136	11	794091496	121
2017	71565152	71565152	12	858781824	144
2018	73473998	73473998	13	955161974	169
2019	74770328	74770328	14	1046784592	196
2020	106431815	106431815	15	1596477225	225
2021	92146685	92146685	16	1474346960	256

```python
ps1.pop("Total")
ps1
```

Year	Y	X	XY	X^2
2006	21245033	1	21245033	1
2007	33633628	2	67267256	4
2008	44453749	3	133361247	9
2009	52264830	4	209059320	16
2010	54933139	5	274665695	25
2011	78074711	6	468448266	36
2012	74959336	7	524715352	49
2013	69262169	8	554097352	64
2014	57685277	9	519167493	81
2015	67650545	10	676505450	100
2016	72190136	11	794091496	121
2017	71565152	12	858781824	144
2018	73473998	13	955161974	169
2019	74770328	14	1046784592	196
2020	106431815	15	1596477225	225
2021	92146685	16	1474346960	256

```python
summation = ps1.sum(axis = 0)
summation = summation.rename('Sum')
summation
```

```
Y        1044740531
X               136
XY      10174176535
X^2            1496
Name: Sum, dtype: int64
```

```python
mean = ps1.mean(axis = 0)
mean = mean.rename('Mean')
mean
```

```
Y       65296283
X              8
XY     635886033
X^2           94
Name: Mean, dtype: float64
```

```
1  ps1 = ps1.append(summation)
2  ps1
```

	Y	X	XY	X^2
Year				
2006	21245033	1	21245033	1
2007	33633628	2	67267256	4
2008	44453749	3	133361247	9
2009	52264830	4	209059320	16
2010	54933139	5	274665695	25
2011	78074711	6	468448266	36
2012	74959336	7	524715352	49
2013	69262169	8	554097352	64
2014	57685277	9	519167493	81
2015	67650545	10	676505450	100
2016	72190136	11	794091496	121
2017	71565152	12	858781824	144
2018	73473998	13	955161974	169
2019	74770328	14	1046784592	196
2020	106431815	15	1596477225	225
2021	92146685	16	1474346960	256
Sum	1044740531	136	10174176535	1496

```
ps1 = ps1.append(mean)
ps1
```

	Y	X	XY	X^2
Year				
2006	21245033	1	21245033	1
2007	33633628	2	67267256	4
2008	44453749	3	133361247	9
2009	52264830	4	209059320	16
2010	54933139	5	274665695	25
2011	78074711	6	468448266	36
2012	74959336	7	524715352	49
2013	69262169	8	554097352	64
2014	57685277	9	519167493	81
2015	67650545	10	676505450	100
2016	72190136	11	794091496	121
2017	71565152	12	858781824	144
2018	73473998	13	955161974	169
2019	74770328	14	1046784592	196
2020	106431815	15	1596477225	225
2021	92146685	16	1474346960	256
Sum	1044740531	136	10174176535	1496
Mean	65296283	8	635886033	94

```
n = len(ps1['X']) - 2
beta = (n * (ps1['XY'].loc['Sum']) - (ps1['X'].loc['Sum'] * ps1['Y'].loc['Sum']))
        / (n * (ps1['X^2'].loc['Sum']) - (ps1['X'].loc['Sum']) ** 2)
print(beta)
```

```
3805535.3573529413
```

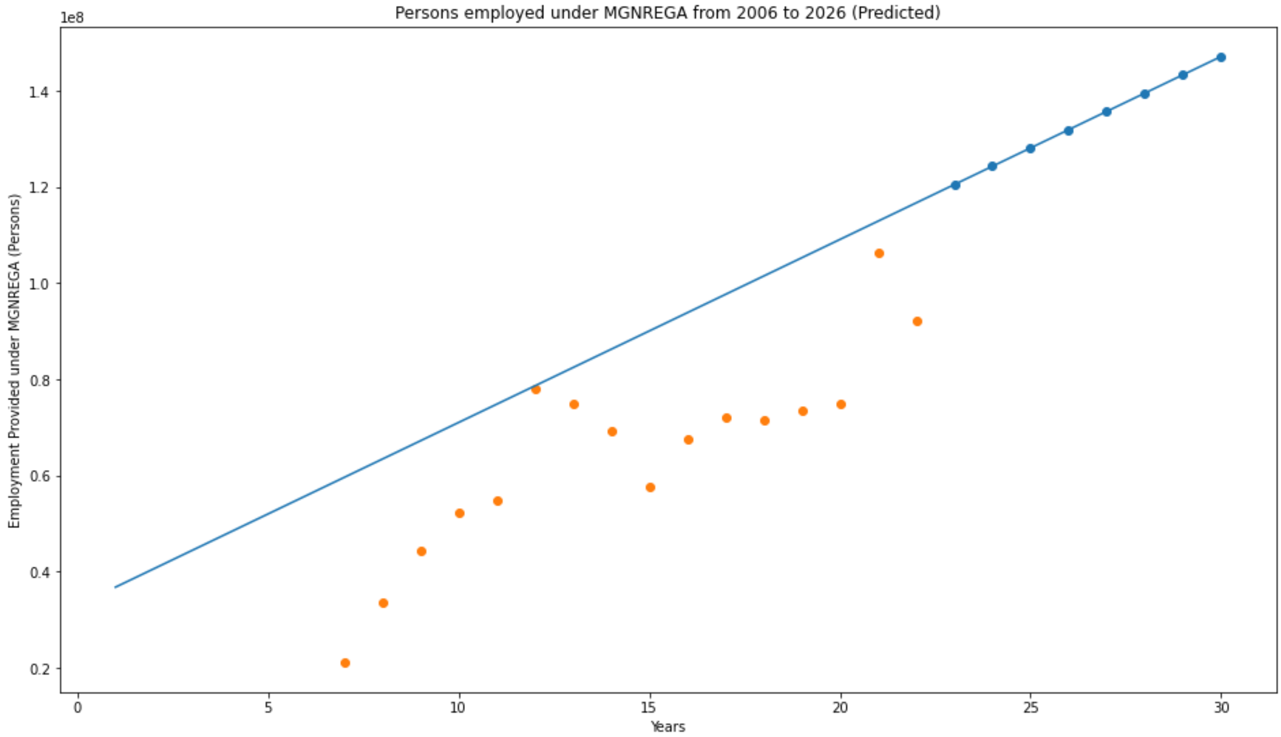
```
def reg(x):
    y = ps1['Y'].loc['Mean'] + beta * (x - ps1['X'].loc['Mean'])
    return y
```

```python
x_values=[]
y_values=[]

for x in range(23,31):
    x_values.append(x)
    y_values.append(reg(x))

predictions = pd.DataFrame(y_values,[i + 1999 for i in x_values])

predictions.columns=['Predicted Employment']
predictions
```

	Predicted Employment
2022	120476546
2023	124282081
2024	128087617
2025	131893152
2026	135698687
2027	139504223
2028	143309758
2029	147115293

```python
import matplotlib.pyplot as plt
plt.figure(figsize=(16,9))
x_values=[]
y_values=[]
for x in range(23,31):
    x_values.append(x)
    y_values.append(reg(x))
plt.scatter([range(23,31)],y_values)
pred=pd.DataFrame(y_values,[i+1999 for i in x_values])
pred.columns=["Predictions"]
print(pred)
temp=[]
for i in ps1_in["Year"]:
    temp.append(int(i)-1999)

plt.scatter(temp,ps1_in['Total'])

x_line=[]
y_line=[]
for x in range(1,31):
    x_line.append(x)
    y_line.append(reg(x))
plt.plot(x_line,y_line)
plt.title("Persons employed under MGNREGA from 2006 to 2026 (Predicted)")
plt.xlabel("Years")
plt.ylabel("Employment Provided under MGNREGA (Persons)")
plt.show()
```

```
      Predictions
2022    120476546
2023    124282081
2024    128087617
2025    131893152
2026    135698687
2027    139504223
2028    143309758
2029    147115293
```



Appendix 1: Objective 1.3

```python
import pandas as pd
mat = pd.read_csv('matrix.csv')
mat
```

	Unnamed: 0	ANDAMAN AND NICOBAR	ANDHRA PRADESH	ARUNACHAL PRADESH	ASSAM	BIHAR	CHHATTISGARH	GOA	GUJARAT	HARYANA	...	ODISHA	PUDUCHERRY	PUNJAB
0	ANDAMAN AND NICOBAR	1	0	0	0	2	0	2	2	2	...	0	2	2
1	ANDHRA PRADESH	2	1	2	2	2	0	2	2	2	...	2	2	2
2	ARUNACHAL PRADESH	2	0	1	2	2	0	2	2	2	...	2	2	2
3	ASSAM	2	0	0	1	2	0	2	2	2	...	0	2	2
4	BIHAR	0	0	0	0	1	0	2	2	2	...	0	0	2
5	CHHATTISGARH	2	2	2	2	2	1	2	2	2	...	2	2	2
6	GOA	0	0	0	0	0	0	1	0	0	...	0	0	0
7	GUJARAT	0	0	0	0	0	0	2	1	2	...	0	0	0
8	HARYANA	0	0	0	0	0	0	2	0	1	...	0	0	0
9	HIMACHAL PRADESH	2	0	0	2	2	0	2	2	2	...	2	2	2
10	JAMMU AND KASHMIR	2	2	2	2	2	2	2	2	2	...	2	2	2
11	JHARKHAND	2	0	0	0	2	0	2	2	2	...	0	2	2
12	KARNATAKA	2	0	0	0	2	0	2	2	2	...	0	2	2
13	KERALA	2	0	0	0	2	0	2	2	2	...	0	2	2
14	LAKSHADWEEP	0	0	0	0	0	0	2	0	0	...	0	0	0
15	MADHYA PRADESH	2	0	0	2	2	0	2	2	2	...	2	2	2
16	MAHARASHTRA	0	0	0	0	0	0	2	2	2	...	0	0	2
17	MANIPUR	2	2	2	2	2	2	2	2	2	...	2	2	2
18	MEGHALAYA	2	2	2	2	2	2	2	2	2	...	2	2	2
19	MIZORAM	2	2	2	2	2	2	2	2	2	...	2	2	2
20	NAGALAND	2	2	2	2	2	2	2	2	2	...	2	2	2
21	ODISHA	2	0	0	2	2	0	2	2	2	...	1	2	2
22	PUDUCHERRY	0	0	0	0	2	0	2	2	2	...	0	1	2
23	PUNJAB	0	0	0	0	0	0	2	2	2	...	0	0	1
24	RAJASTHAN	2	0	0	2	2	0	2	2	2	...	2	2	2
25	SIKKIM	2	0	2	2	2	0	2	2	2	...	2	2	2
26	TAMIL NADU	2	0	0	2	2	0	2	2	2	...	2	2	2
27	TRIPURA	2	2	2	2	2	2	2	2	2	...	2	2	2
28	UTTAR PRADESH	2	0	0	0	2	0	2	2	2	...	0	2	2
29	UTTARAKHAND	2	0	0	0	2	0	2	2	2	...	0	2	2
30	WEST BENGAL	2	0	0	2	2	0	2	2	2	...	2	2	2

31 rows × 32 columns

```python
mat_A = mat.iloc[:,1:].to_numpy()
print(mat_A)
```

```
[[1 0 0 0 2 0 2 2 2 0 0 0 0 0 2 0 2 0 0 0 0 0 2 2 0 0 0 0 0 0 0]
 [2 1 2 2 2 0 2 2 2 2 0 2 2 2 2 2 2 0 0 0 0 2 2 2 2 2 2 0 2 2 2]
 [2 0 1 2 2 0 2 2 2 2 0 2 2 2 2 2 2 0 0 0 0 2 2 2 2 0 2 0 2 2 2]
 [2 0 0 1 2 0 2 2 2 0 0 2 2 2 2 0 2 0 0 0 0 0 2 2 0 0 0 0 2 2 0]
 [0 0 0 0 1 0 2 2 2 0 0 0 0 0 2 0 2 0 0 0 0 0 0 2 0 0 0 0 0 0 0]
 [2 2 2 2 2 1 2 2 2 2 0 2 2 2 2 2 2 0 0 0 0 2 2 2 2 2 2 0 2 2 2]
 [0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0]
 [0 0 0 0 0 0 2 1 2 0 0 0 0 0 2 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0]
 [0 0 0 0 0 0 2 0 1 0 0 0 0 0 2 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0]
 [2 0 0 2 2 0 2 2 2 1 0 2 2 2 2 2 2 0 0 0 0 2 2 2 0 0 0 0 2 2 2]
 [2 2 2 2 2 2 2 2 2 2 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2]
 [2 0 0 0 2 0 2 2 2 0 0 1 0 2 2 0 2 0 0 0 0 0 2 2 0 0 0 0 2 2 0]
 [2 0 0 0 2 0 2 2 2 0 0 2 1 2 2 0 2 0 0 0 0 0 2 2 0 0 0 0 2 2 0]
 [2 0 0 0 2 0 2 2 2 0 0 0 0 1 2 0 2 0 0 0 0 0 2 2 0 0 0 0 2 0 0]
 [0 0 0 0 0 0 2 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0]
 [2 0 0 2 2 0 2 2 2 0 0 2 2 2 2 1 2 0 0 0 0 2 2 2 0 0 0 0 2 2 0]
 [0 0 0 0 0 0 2 2 2 0 0 0 0 0 2 0 1 0 0 0 0 0 0 2 0 0 0 0 0 0 0]
 [2 2 2 2 2 2 2 2 2 2 0 2 2 2 2 2 2 1 2 2 0 2 2 2 2 2 2 0 2 2 2]
 [2 2 2 2 2 2 2 2 2 2 0 2 2 2 2 2 2 0 1 2 0 2 2 2 2 2 2 0 2 2 2]
 [2 2 2 2 2 2 2 2 2 2 0 2 2 2 2 2 2 0 0 1 0 2 2 2 2 2 2 0 2 2 2]
 [2 2 2 2 2 2 2 2 2 2 0 2 2 2 2 2 2 2 2 2 1 2 2 2 2 2 2 0 2 2 2]
 [2 0 0 2 2 0 2 2 2 0 0 2 2 2 2 0 2 0 0 0 0 1 2 2 0 0 0 0 2 2 0]
 [0 0 0 0 2 0 2 2 2 0 0 0 0 0 2 0 2 0 0 0 0 0 1 2 0 0 0 0 0 0 0]
 [0 0 0 0 0 0 2 2 2 0 0 0 0 0 2 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0]
 [2 0 0 2 2 0 2 2 2 2 0 2 2 2 2 2 2 0 0 0 0 2 2 2 1 0 0 0 2 2 2]
 [2 0 2 2 2 0 2 2 2 2 0 2 2 2 2 2 2 0 0 0 0 2 2 2 2 1 2 0 2 2 2]
 [2 0 0 2 2 0 2 2 2 2 0 2 2 2 2 2 2 0 0 0 0 2 2 2 2 0 1 0 2 2 2]
 [2 2 2 2 2 2 2 2 2 2 0 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 1 2 2 2]
 [2 0 0 0 2 0 2 2 2 0 0 0 0 0 2 0 2 0 0 0 0 0 2 2 0 0 0 0 1 0 0]
 [2 0 0 0 2 0 2 2 2 0 0 0 0 2 2 0 2 0 0 0 0 0 2 2 0 0 0 0 2 1 0]
 [2 0 0 2 2 0 2 2 2 0 0 2 2 2 2 2 2 0 0 0 0 2 2 2 0 0 0 0 2 2 1]]
```

```python
import numpy as np
X = []
eigen=[0]

X.append(np.array([[1],[1],[1],[1],[1],[1],[1],[1],[1],[1],[1],[1],[1],[1],[1],[1],[1],[1],[1],[1],[1],[1],[1],[1],[1],[1],[
print(mat_A)
print('\n')
print(X)
print('\n')
i=0
while len(eigen)==1 or np.round(eigen[i],3)!=np.round(eigen[i-1],3):
    X.append(np.matmul(mat_A,X[i]))
    eigen.append(max(X[i+1]))
    X[i+1]= np.round(X[i+1]/eigen[i+1],4)
    print("Iteration",i+1,":")
    print(X[i+1],"\t",eigen[i+1])
    print()
    i+=1
```

```
[[1 0 0 0 2 0 2 2 2 0 0 0 0 0 2 0 2 0 0 0 0 0 2 2 0 0 0 0 0 0 0]
 [2 1 2 2 2 0 2 2 2 2 0 2 2 2 2 2 2 0 0 0 0 2 2 2 2 2 2 0 2 2 2]
 [2 0 1 2 2 0 2 2 2 2 0 2 2 2 2 2 2 0 0 0 0 2 2 2 2 0 2 0 2 2 2]
 [2 0 0 1 2 0 2 2 2 0 0 2 2 2 2 0 2 0 0 0 0 0 2 2 0 0 0 0 2 2 0]
 [0 0 0 0 1 0 2 2 2 0 0 0 0 0 2 0 2 0 0 0 0 0 0 2 0 0 0 0 0 0 0]
 [2 2 2 2 2 1 2 2 2 2 0 2 2 2 2 2 2 0 0 0 0 2 2 2 2 2 2 0 2 2 2]
 [0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0]
 [0 0 0 0 0 0 2 1 2 0 0 0 0 0 2 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0]
 [0 0 0 0 0 0 2 0 1 0 0 0 0 0 2 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0]
 [2 0 0 2 2 0 2 2 2 1 0 2 2 2 2 2 2 0 0 0 0 2 2 2 0 0 0 0 2 2 2]
 [2 2 2 2 2 2 2 2 2 2 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2]
 [2 0 0 0 2 0 2 2 2 0 0 1 0 2 2 0 2 0 0 0 0 0 2 2 0 0 0 0 2 2 0]
 [2 0 0 0 2 0 2 2 2 0 0 2 1 2 2 0 2 0 0 0 0 0 2 2 0 0 0 0 2 2 0]
 [2 0 0 0 2 0 2 2 2 0 0 0 0 1 2 0 2 0 0 0 0 0 2 2 0 0 0 0 2 0 0]
 [0 0 0 0 0 0 2 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0]
 [2 0 0 2 2 0 2 2 2 0 0 2 2 2 2 1 2 0 0 0 0 2 2 2 0 0 0 0 2 2 0]
 [0 0 0 0 0 0 2 2 2 0 0 0 0 0 2 0 1 0 0 0 0 0 2 0 0 0 0 0 0 0 0]
 [2 2 2 2 2 2 2 2 2 2 0 2 2 2 2 2 2 1 2 2 0 2 2 2 2 2 2 0 2 2 2]
 [2 2 2 2 2 2 2 2 2 2 0 2 2 2 2 2 2 0 1 2 0 2 2 2 2 2 2 0 2 2 2]
 [2 2 2 2 2 2 2 2 2 2 0 2 2 2 2 2 2 0 0 1 0 2 2 2 2 2 2 0 2 2 2]
 [2 2 2 2 2 2 2 2 2 2 0 2 2 2 2 2 2 2 2 2 1 2 2 2 2 2 2 0 2 2 2]
 [2 0 0 2 2 0 2 2 2 0 0 2 2 2 2 0 2 0 0 0 0 1 2 2 0 0 0 0 2 2 0]
 [0 0 0 0 2 0 2 2 2 0 0 0 0 0 2 0 2 0 0 0 0 0 1 2 0 0 0 0 0 0 0]
 [0 0 0 0 0 0 2 2 2 0 0 0 0 0 2 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0]
 [2 0 0 2 2 0 2 2 2 2 0 2 2 2 2 2 2 0 0 0 0 2 2 2 1 0 0 0 2 2 2]
 [2 0 2 2 2 0 2 2 2 2 0 2 2 2 2 2 2 0 0 0 0 2 2 2 2 1 2 0 2 2 2]
 [2 0 0 2 2 0 2 2 2 2 0 2 2 2 2 2 2 0 0 0 0 2 2 2 2 0 1 0 2 2 2]
 [2 2 2 2 2 2 2 2 2 2 0 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 1 2 2 2]
 [2 0 0 0 2 0 2 2 2 0 0 0 0 0 2 0 2 0 0 0 0 0 2 2 0 0 0 0 1 0 0]
 [2 0 0 0 2 0 2 2 2 0 0 0 0 2 2 0 2 0 0 0 0 0 2 2 0 0 0 0 2 1 0]
 [2 0 0 2 2 0 2 2 2 0 0 2 2 2 2 2 2 0 0 0 0 2 2 2 0 0 0 0 2 2 1]]
```

```
[array([[1],
       [1],
       [1],
       [1],
       [1],
       [1],
       [1],
       [1],
       [1],
       [1],
       [1],
       [1],
       [1],
       [1],
       [1],
       [1],
       [1],
       [1],
       [1],
       [1],
       [1],
       [1],
       [1],
       [1],
       [1],
       [1],
       [1],
       [1],
       [1],
       [1],
       [1]])]
```

```
Iteration 1 :
[[0.2787]
 [0.7705]
 [0.7049]
 [0.4754]
 [0.2131]
 [0.8033]
 [0.0164]
 [0.1148]
 [0.082 ]
 [0.6066]
 [1.    ]
 [0.4098]
 [0.4426]
 [0.3443]
 [0.0492]
 [0.541 ]
 [0.1803]
 [0.9016]
 [0.8689]
 [0.8361]
 [0.9344]
 [0.5082]
 [0.2459]
 [0.1475]
 [0.6393]
 [0.7377]
 [0.6721]
 [0.9672]
 [0.3115]
 [0.377 ]
 [0.5738]]        [61]
```

```
Iteration 2 :
[[7.790e-02]
 [5.938e-01]
 [4.970e-01]
 [2.262e-01]
 [4.570e-02]
 [6.454e-01]
 [5.000e-04]
 [1.340e-02]
 [7.000e-03]
 [3.681e-01]
 [1.000e+00]
 [1.682e-01]
 [1.961e-01]
 [1.188e-01]
 [2.700e-03]
 [2.929e-01]
 [3.280e-02]
 [8.130e-01]
 [7.550e-01]
 [6.991e-01]
 [8.732e-01]
 [2.585e-01]
 [6.070e-02]
 [2.200e-02]
 [4.089e-01]
 [5.443e-01]
 [4.519e-01]
 [9.355e-01]
 [9.730e-02]
 [1.424e-01]
 [3.294e-01]]        [30.5082]
```

```
Iteration 3 :
[[2.200e-02]
 [4.578e-01]
 [3.508e-01]
 [1.079e-01]
 [9.900e-03]
 [5.187e-01]
 [0.000e+00]
 [1.700e-03]
 [7.000e-04]
 [2.237e-01]
 [1.000e+00]
 [6.930e-02]
 [8.720e-02]
 [4.120e-02]
 [2.000e-04]
 [1.588e-01]
 [6.100e-03]
 [7.332e-01]
 [6.562e-01]
 [5.848e-01]
 [8.161e-01]
 [1.318e-01]
 [1.520e-02]
 [3.400e-03]
 [2.619e-01]
 [4.019e-01]
 [3.041e-01]
 [9.049e-01]
 [3.060e-02]
 [5.400e-02]
 [1.894e-01]]        [20.3554]
```

```
Iteration 4 :
[[6.300e-03]
 [3.533e-01]
 [2.479e-01]
 [5.170e-02]
 [2.200e-03]
 [4.172e-01]
 [0.000e+00]
 [2.000e-04]
 [1.000e-04]
 [1.362e-01]
 [1.000e+00]
 [2.870e-02]
 [3.900e-02]
 [1.440e-02]
 [0.000e+00]
 [8.640e-02]
 [1.200e-03]
 [6.615e-01]
 [5.706e-01]
 [4.894e-01]
 [7.628e-01]
 [6.740e-02]
 [3.900e-03]
 [6.000e-04]
 [1.680e-01]
 [2.971e-01]
 [2.050e-01]
 [8.754e-01]
 [9.700e-03]
 [2.070e-02]
 [1.092e-01]]        [15.287]
```

```
Iteration 5 :
[[1.900e-03]
 [2.730e-01]
 [1.755e-01]
 [2.500e-02]
 [5.000e-04]
 [3.359e-01]
 [0.000e+00]
 [0.000e+00]
 [0.000e+00]
 [8.320e-02]
 [1.000e+00]
 [1.200e-02]
 [1.750e-02]
 [5.100e-03]
 [0.000e+00]
 [4.720e-02]
 [2.000e-04]
 [5.970e-01]
 [4.964e-01]
 [4.099e-01]
 [7.132e-01]
 [3.470e-02]
 [1.000e-03]
 [1.000e-04]
 [1.080e-01]
 [2.199e-01]
 [1.385e-01]
 [8.469e-01]
 [3.200e-03]
 [8.000e-03]
 [6.320e-02]]        [12.2522]
```

```
Iteration 6 :
[[5.000e-04]
 [2.113e-01]
 [1.245e-01]
 [1.210e-02]
 [1.000e-04]
 [2.708e-01]
 [0.000e+00]
 [0.000e+00]
 [0.000e+00]
 [5.100e-02]
 [1.000e+00]
 [5.100e-03]
 [8.000e-03]
 [1.800e-03]
 [0.000e+00]
 [2.600e-02]
 [0.000e+00]
 [5.391e-01]
 [4.322e-01]
 [3.437e-01]
 [6.671e-01]
 [1.790e-02]
 [3.000e-04]
 [0.000e+00]
 [6.970e-02]
 [1.631e-01]
 [9.380e-02]
 [8.195e-01]
 [1.000e-03]
 [3.100e-03]
 [3.670e-02]]        [10.234]
```

```
Iteration 7 :
[[1.000e-04]
 [1.638e-01]
 [8.850e-02]
 [5.900e-03]
 [0.000e+00]
 [2.186e-01]
 [0.000e+00]
 [0.000e+00]
 [0.000e+00]
 [3.140e-02]
 [1.000e+00]
 [2.100e-03]
 [3.600e-03]
 [6.000e-04]
 [0.000e+00]
 [1.430e-02]
 [0.000e+00]
 [4.871e-01]
 [3.766e-01]
 [2.884e-01]
 [6.242e-01]
 [9.300e-03]
 [1.000e-04]
 [0.000e+00]
 [4.510e-02]
 [1.212e-01]
 [6.370e-02]
 [7.932e-01]
 [3.000e-04]
 [1.200e-03]
 [2.140e-02]]        [8.7968]
```

```
Iteration 8 :
[[0.000e+00]
 [1.271e-01]
 [6.300e-02]
 [2.800e-03]
 [0.000e+00]
 [1.766e-01]
 [0.000e+00]
 [0.000e+00]
 [0.000e+00]
 [1.930e-02]
 [1.000e+00]
 [9.000e-04]
 [1.600e-03]
 [2.000e-04]
 [0.000e+00]
 [7.900e-03]
 [0.000e+00]
 [4.403e-01]
 [3.284e-01]
 [2.423e-01]
 [5.842e-01]
 [4.800e-03]
 [0.000e+00]
 [0.000e+00]
 [2.920e-02]
 [9.020e-02]
 [4.330e-02]
 [7.678e-01]
 [1.000e-04]
 [4.000e-04]
 [1.250e-02]]        [7.7214]
```

```
Iteration 9 :
[[0.000e+00]
 [9.870e-02]
 [4.490e-02]
 [1.300e-03]
 [0.000e+00]
 [1.428e-01]
 [0.000e+00]
 [0.000e+00]
 [0.000e+00]
 [1.190e-02]
 [1.000e+00]
 [3.000e-04]
 [7.000e-04]
 [1.000e-04]
 [0.000e+00]
 [4.300e-03]
 [0.000e+00]
 [3.981e-01]
 [2.865e-01]
 [2.036e-01]
 [5.469e-01]
 [2.400e-03]
 [0.000e+00]
 [0.000e+00]
 [1.890e-02]
 [6.710e-02]
 [2.940e-02]
 [7.433e-01]
 [0.000e+00]
 [1.000e-04]
 [7.200e-03]]        [6.8858]
```

```
Iteration 10 :
[[0.000e+00]
 [7.650e-02]
 [3.190e-02]
 [6.000e-04]
 [0.000e+00]
 [1.154e-01]
 [0.000e+00]
 [0.000e+00]
 [0.000e+00]
 [7.200e-03]
 [1.000e+00]
 [1.000e-04]
 [3.000e-04]
 [0.000e+00]
 [0.000e+00]
 [2.300e-03]
 [0.000e+00]
 [3.601e-01]
 [2.499e-01]
 [1.711e-01]
 [5.121e-01]
 [1.200e-03]
 [0.000e+00]
 [0.000e+00]
 [1.210e-02]
 [4.990e-02]
 [1.990e-02]
 [7.196e-01]
 [0.000e+00]
 [0.000e+00]
 [4.100e-03]]        [6.217]
```

```
Iteration 11 :
[[0.000e+00]
 [5.920e-02]
 [2.250e-02]
 [2.000e-04]
 [0.000e+00]
 [9.310e-02]
 [0.000e+00]
 [0.000e+00]
 [0.000e+00]
 [4.300e-03]
 [1.000e+00]
 [0.000e+00]
 [1.000e-04]
 [0.000e+00]
 [0.000e+00]
 [1.200e-03]
 [0.000e+00]
 [3.255e-01]
 [2.179e-01]
 [1.436e-01]
 [4.794e-01]
 [6.000e-04]
 [0.000e+00]
 [0.000e+00]
 [7.700e-03]
 [3.690e-02]
 [1.340e-02]
 [6.966e-01]
 [0.000e+00]
 [0.000e+00]
 [2.300e-03]]     [5.6686]
```

```
Iteration 12 :
[[0.000e+00]
 [4.560e-02]
 [1.580e-02]
 [1.000e-04]
 [0.000e+00]
 [7.490e-02]
 [0.000e+00]
 [0.000e+00]
 [0.000e+00]
 [2.500e-03]
 [1.000e+00]
 [0.000e+00]
 [0.000e+00]
 [0.000e+00]
 [0.000e+00]
 [6.000e-04]
 [0.000e+00]
 [2.940e-01]
 [1.897e-01]
 [1.203e-01]
 [4.485e-01]
 [2.000e-04]
 [0.000e+00]
 [0.000e+00]
 [4.800e-03]
 [2.720e-02]
 [8.900e-03]
 [6.743e-01]
 [0.000e+00]
 [0.000e+00]
 [1.200e-03]]     [5.209]
```

```
Iteration 13 :
[[0.000e+00]
 [3.490e-02]
 [1.090e-02]
 [0.000e+00]
 [0.000e+00]
 [5.990e-02]
 [0.000e+00]
 [0.000e+00]
 [0.000e+00]
 [1.400e-03]
 [1.000e+00]
 [0.000e+00]
 [0.000e+00]
 [0.000e+00]
 [0.000e+00]
 [2.000e-04]
 [0.000e+00]
 [2.652e-01]
 [1.648e-01]
 [1.005e-01]
 [4.194e-01]
 [1.000e-04]
 [0.000e+00]
 [0.000e+00]
 [2.900e-03]
 [1.980e-02]
 [5.800e-03]
 [6.524e-01]
 [0.000e+00]
 [0.000e+00]
 [6.000e-04]]     [4.8172]
```

```
Iteration 14 :
[[0.000e+00]
 [2.640e-02]
 [7.300e-03]
 [0.000e+00]
 [0.000e+00]
 [4.760e-02]
 [0.000e+00]
 [0.000e+00]
 [0.000e+00]
 [7.000e-04]
 [1.000e+00]
 [0.000e+00]
 [0.000e+00]
 [0.000e+00]
 [0.000e+00]
 [1.000e-04]
 [0.000e+00]
 [2.387e-01]
 [1.427e-01]
 [8.340e-02]
 [3.916e-01]
 [0.000e+00]
 [0.000e+00]
 [0.000e+00]
 [1.700e-03]
 [1.420e-02]
 [3.600e-03]
 [6.310e-01]
 [0.000e+00]
 [0.000e+00]
 [3.000e-04]]     [4.4776]
```

```
Iteration 15 :
[[0.000e+00]
 [1.970e-02]
 [4.800e-03]
 [0.000e+00]
 [0.000e+00]
 [3.740e-02]
 [0.000e+00]
 [0.000e+00]
 [0.000e+00]
 [4.000e-04]
 [1.000e+00]
 [0.000e+00]
 [0.000e+00]
 [0.000e+00]
 [0.000e+00]
 [0.000e+00]
 [0.000e+00]
 [2.141e-01]
 [1.228e-01]
 [6.870e-02]
 [3.650e-01]
 [0.000e+00]
 [0.000e+00]
 [0.000e+00]
 [9.000e-04]
 [1.000e-02]
 [2.200e-03]
 [6.097e-01]
 [0.000e+00]
 [0.000e+00]
 [1.000e-04]]     [4.1786]
```

```
Iteration 16 :
[[0.000e+00]
 [1.440e-02]
 [3.100e-03]
 [0.000e+00]
 [0.000e+00]
 [2.900e-02]
 [0.000e+00]
 [0.000e+00]
 [0.000e+00]
 [2.000e-04]
 [1.000e+00]
 [0.000e+00]
 [0.000e+00]
 [0.000e+00]
 [0.000e+00]
 [0.000e+00]
 [0.000e+00]
 [1.913e-01]
 [1.051e-01]
 [5.620e-02]
 [3.393e-01]
 [0.000e+00]
 [0.000e+00]
 [0.000e+00]
 [5.000e-04]
 [6.900e-03]
 [1.300e-03]
 [5.885e-01]
 [0.000e+00]
 [0.000e+00]
 [0.000e+00]]     [3.9116]
```

```
Iteration 17 :
[[0.000e+00]
 [1.050e-02]
 [1.900e-03]
 [0.000e+00]
 [0.000e+00]
 [2.230e-02]
 [0.000e+00]
 [0.000e+00]
 [0.000e+00]
 [1.000e-04]
 [1.000e+00]
 [0.000e+00]
 [0.000e+00]
 [0.000e+00]
 [0.000e+00]
 [0.000e+00]
 [0.000e+00]
 [1.701e-01]
 [8.940e-02]
 [4.550e-02]
 [3.147e-01]
 [0.000e+00]
 [0.000e+00]
 [0.000e+00]
 [2.000e-04]
 [4.700e-03]
 [7.000e-04]
 [5.674e-01]
 [0.000e+00]
 [0.000e+00]
 [0.000e+00]]     [3.6716]
```

```
Iteration 18 :
[[0.000e+00]
 [7.400e-03]
 [1.100e-03]
 [0.000e+00]
 [0.000e+00]
 [1.690e-02]
 [0.000e+00]
 [0.000e+00]
 [0.000e+00]
 [0.000e+00]
 [1.000e+00]
 [0.000e+00]
 [0.000e+00]
 [0.000e+00]
 [0.000e+00]
 [0.000e+00]
 [0.000e+00]
 [1.507e-01]
 [7.560e-02]
 [3.660e-02]
 [2.910e-01]
 [0.000e+00]
 [0.000e+00]
 [0.000e+00]
 [1.000e-04]
 [3.000e-03]
 [4.000e-04]
 [5.463e-01]
 [0.000e+00]
 [0.000e+00]
 [0.000e+00]]     [3.455]
```

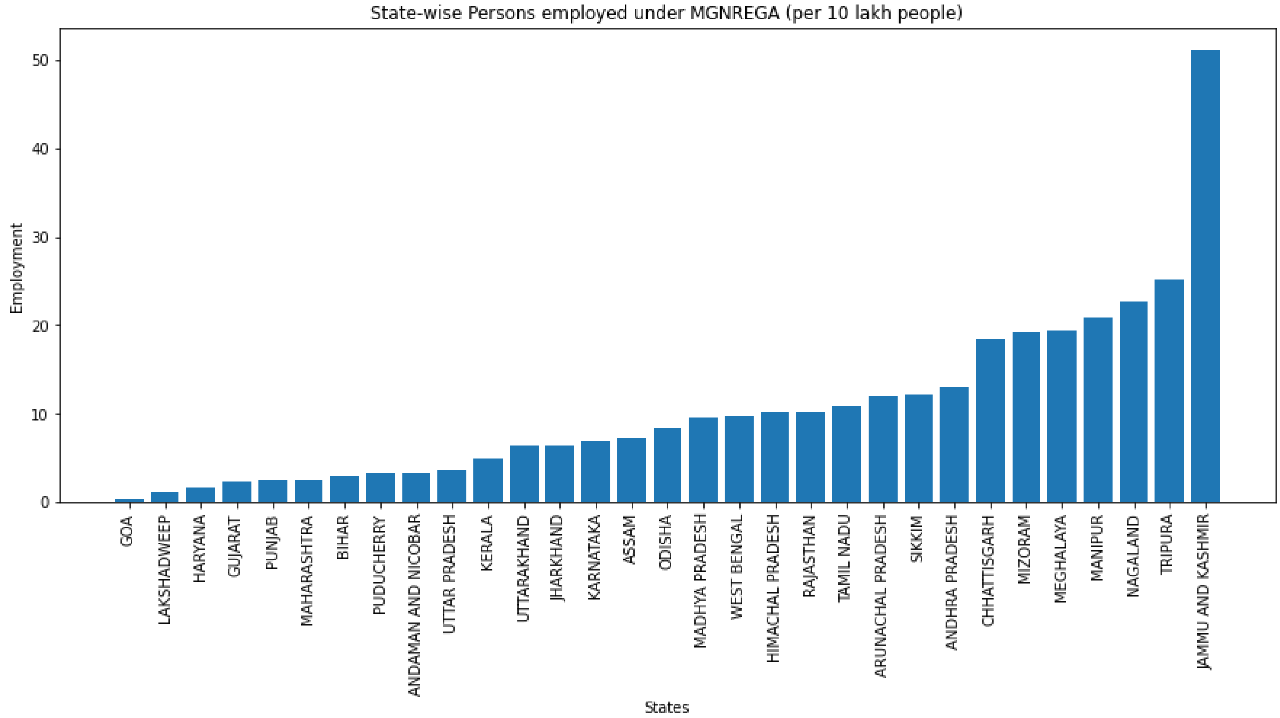
```
Iteration 19 :
[[0.000e+00]
 [5.100e-03]
 [6.000e-04]
 [0.000e+00]
 [0.000e+00]
 [1.260e-02]
 [0.000e+00]
 [0.000e+00]
 [0.000e+00]
 [0.000e+00]
 [1.000e+00]
 [0.000e+00]
 [0.000e+00]
 [0.000e+00]
 [0.000e+00]
 [0.000e+00]
 [0.000e+00]
 [1.329e-01]
 [6.340e-02]
 [2.900e-02]
 [2.684e-01]
 [0.000e+00]
 [0.000e+00]
 [0.000e+00]
 [0.000e+00]
 [1.900e-03]
 [2.000e-04]
 [5.254e-01]
 [0.000e+00]
 [0.000e+00]
 [0.000e+00]]     [3.2582]
```

```
Iteration 20 :
[[0.000e+00]
 [3.400e-03]
 [3.000e-04]
 [0.000e+00]
 [0.000e+00]
 [9.200e-03]
 [0.000e+00]
 [0.000e+00]
 [0.000e+00]
 [0.000e+00]
 [1.000e+00]
 [0.000e+00]
 [0.000e+00]
 [0.000e+00]
 [0.000e+00]
 [0.000e+00]
 [0.000e+00]
 [1.164e-01]
 [5.270e-02]
 [2.270e-02]
 [2.468e-01]
 [0.000e+00]
 [0.000e+00]
 [0.000e+00]
 [0.000e+00]
 [1.100e-03]
 [1.000e-04]
 [5.046e-01]
 [0.000e+00]
 [0.000e+00]
 [0.000e+00]]     [3.079]
```

```
Iteration 21 :
[[0.000e+00]
 [2.200e-03]
 [2.000e-04]
 [0.000e+00]
 [0.000e+00]
 [6.500e-03]
 [0.000e+00]
 [0.000e+00]
 [0.000e+00]
 [0.000e+00]
 [1.000e+00]
 [0.000e+00]
 [0.000e+00]
 [0.000e+00]
 [0.000e+00]
 [0.000e+00]
 [0.000e+00]
 [1.014e-01]
 [4.330e-02]
 [1.750e-02]
 [2.260e-01]
 [0.000e+00]
 [0.000e+00]
 [0.000e+00]
 [0.000e+00]
 [7.000e-04]
 [0.000e+00]
 [4.838e-01]
 [0.000e+00]
 [0.000e+00]
 [0.000e+00]]     [2.9146]
```

```
Iteration 22 :
[[0.000e+00]
 [1.400e-03]
 [1.000e-04]
 [0.000e+00]
 [0.000e+00]
 [4.600e-03]
 [0.000e+00]
 [0.000e+00]
 [0.000e+00]
 [0.000e+00]
 [1.000e+00]
 [0.000e+00]
 [0.000e+00]
 [0.000e+00]
 [0.000e+00]
 [0.000e+00]
 [0.000e+00]
 [8.770e-02]
 [3.530e-02]
 [1.330e-02]
 [2.061e-01]
 [0.000e+00]
 [0.000e+00]
 [0.000e+00]
 [0.000e+00]
 [4.000e-04]
 [0.000e+00]
 [4.630e-01]
 [0.000e+00]
 [0.000e+00]
 [0.000e+00]]     [2.7632]
```

```python
import matplotlib.pyplot as plt
plt.figure(figsize=(15,6))
plt.bar(ps3_sort["States"],ps3_sort["Rank"])
plt.title("State-wise Persons employed under MGNREGA (per 10 lakh people)")
plt.xticks(rotation = 90)
plt.xlabel("States")
plt.ylabel("Employment")
plt.show()
```



Appendix 2:

Objective 2.1

In [1]:

```python
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
```

In [2]:

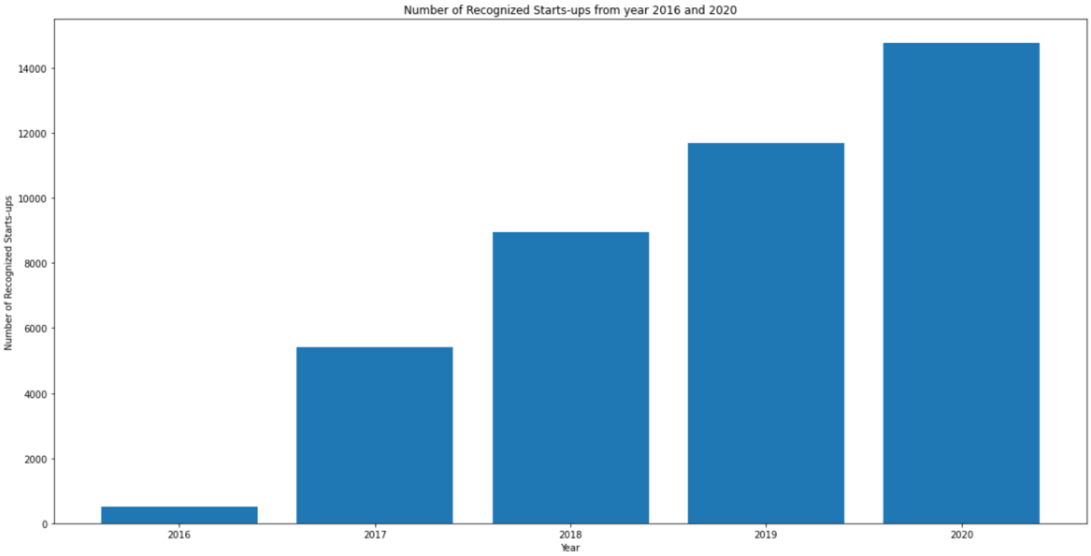
```python
ps1=pd.read_csv("ps1.csv")
ps1
```

Out[2]:

	Calander Year	Recognized Starts-ups	unemployment Rate
0	2016	504	5.662
1	2017	5411	5.549
2	2018	8944	5.370
3	2019	11688	5.371
4	2020	14770	6.469

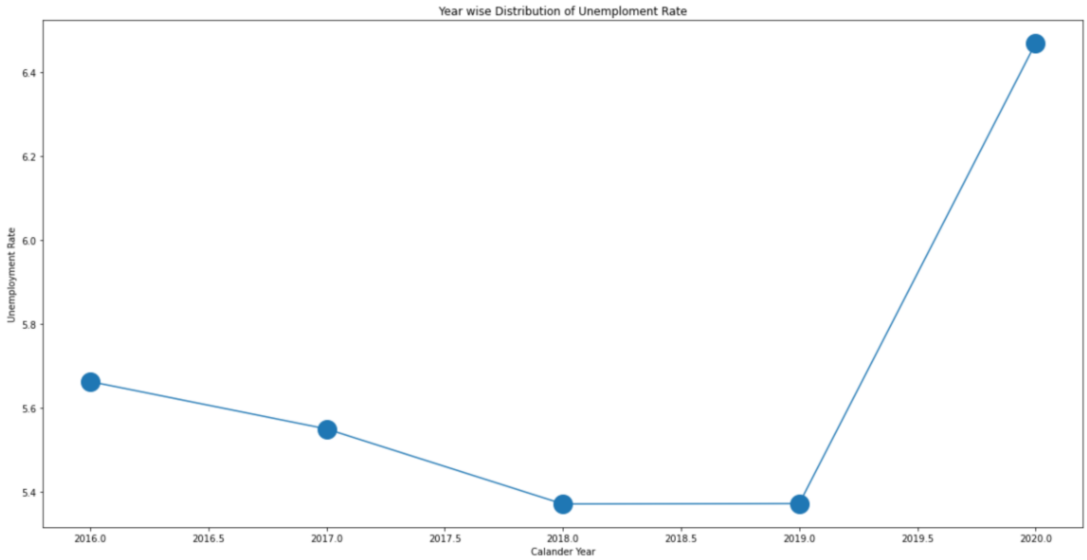
In [3]:

```python
fig=plt.figure(figsize=(20,10))
plt.bar(ps1['Calander Year'], ps1['Recognized Starts-ups'])
plt.xlabel('Year')
plt.ylabel('Number of Recognized Starts-ups')
plt.title('Number of Recognized Starts-ups from year 2016 and 2020')
plt.show()
```



In [12]:

```python
fig=plt.figure(figsize=(20,10))
plt.plot(ps1['Calander Year'],ps1['unemployment Rate'],marker='o', markersize=20)
plt.xlabel('Calander Year')
plt.ylabel('Unemployment Rate')
plt.title('Year wise Distribution of Unemploment Rate')
plt.show()
```



In [4]:
```
correlationcoefficient1=np.corrcoef(ps1['Recognized Starts-ups'],ps1['unemployment Rate'])
correlationcoefficient1
print("Corelation coefficient between the Recognized Starts-ups and GDP per capita(in $) is",correlationcoefficient1[0][1])
```

```
Corelation coefficient between the Recognized Starts-ups and GDP per capita(in $) is 0.43311675394123045
```

In [5]:
```
def CR(r):
    r=float(r)
    if r>-0.1 and r<0.1:
        print("No Correlation")
    elif r>=0.1 and r<0.5:
        print("Weakly Positive Correlation")
    elif r>=0.5 and r<0.8:
        print("Moderately Positive Correlation")
    elif r>=0.8 and r<1.0:
        print("Strongly Positive Correlation")
    elif r>=-0.1 and r<-0.5:
        print("Weekly Negative Correlation")
    elif r>=-0.5 and r<-0.8:
        print("Moderate Negative Correlation")
    elif r>=-0.8 and r<-1.0:
        print("Moderate Negative Correlation")
CR(correlationcoefficient1[0][1])
```

```
Weakly Positive Correlation
```

In [6]:
```
deltar=((correlationcoefficient1[0][1])*((3)**1/2))/((1-(correlationcoefficient1[0][1])**2)**1/2)
deltar
```

Out[6]:
```
1.5993777253096053
```

In [11]:
```
talpha=2.132
if deltar>talpha or deltar<talpha:
    print("null hypothesis is rejected")
else:
    print("null hypothesis is accepted")
```

```
null hypothesis is rejected
```

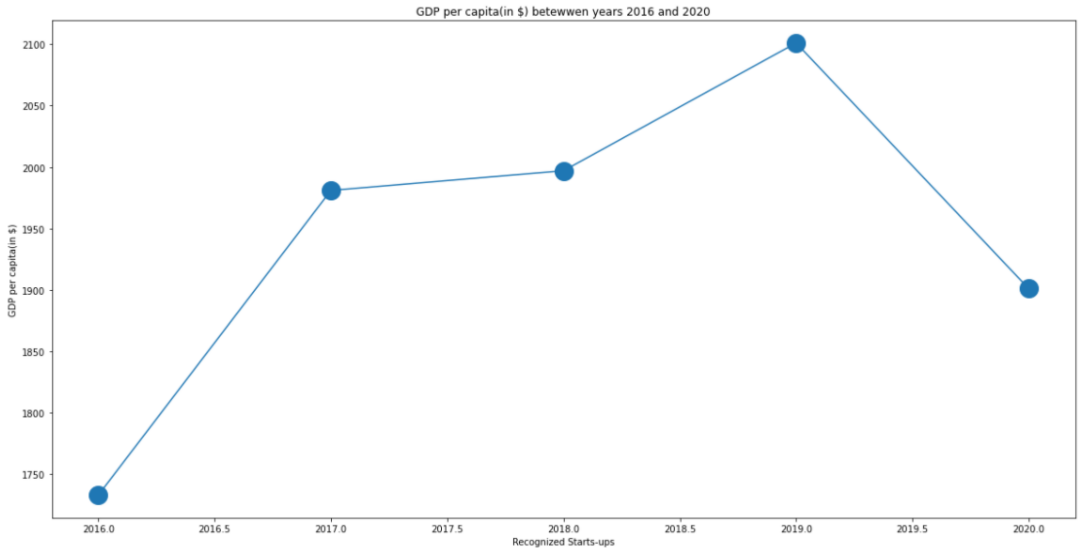
Objective 2.2:

In [16]:
```
ps2=pd.read_csv("ps2.csv")
ps2
```

Out[16]:

	Calander Year	Recognized Starts-ups	GDP per capita(in $)
0	2016	504	1733
1	2017	5411	1981
2	2018	8944	1997
3	2019	11688	2101
4	2020	14770	1901

In [17]:
```
fig=plt.figure(figsize=(20,10))
plt.plot(ps2['Calander Year'],ps2['GDP per capita(in $)'],marker='o', markersize=20)
plt.xlabel('Recognized Starts-ups')
plt.ylabel('GDP per capita(in $)')
plt.title('GDP per capita(in $) betewwen years 2016 and 2020')
plt.show()
```



In [18]:
```
correlationcoefficient=np.corrcoef(ps2['Recognized Starts-ups'],ps2['GDP per capita(in $)'])
correlationcoefficient
print("Corelation coefficient between the Recognized Starts-ups and GDP per capita(in $) is",correlationcoefficient[0][1])
```

```
Corelation coefficient between the Recognized Starts-ups and GDP per capita(in $) is 0.600646417758553
```

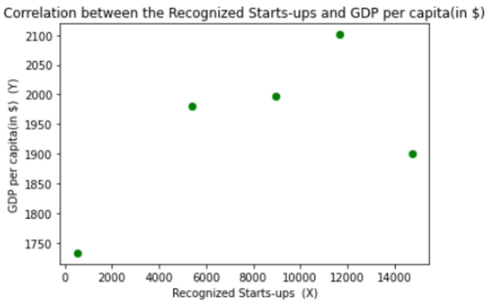
In [19]:
```
def CR(r):
    r=float(r)
    if r>-0.1 and r<0.1:
        print("No Correlation")
    elif r>=0.1 and r<0.5:
        print("Weekly Positive Correlation")
    elif r>=0.5 and r<0.8:
        print("Moderately Positive Correlation")
    elif r>=0.8 and r<1.0:
        print("Strongly Positive Correlation")
    elif r>=-0.1 and r<-0.5:
        print("Weekly Negative Correlation")
    elif r>=-0.5 and r<-0.8:
        print("Moderate Negative Correlation")
    elif r>=-0.8 and r<-1.0:
        print("Moderate Negative Correlation")
```

In [20]:
```
CR(correlationcoefficient[0][1])
```

```
Moderately Positive Correlation
```

In [21]:
```
plt.scatter(ps2['Recognized Starts-ups'], ps2['GDP per capita(in $)'], label= "stars", color= "green",
            marker= "o", s=40)
plt.title('Correlation between the Recognized Starts-ups and GDP per capita(in $)')
plt.xlabel('Recognized Starts-ups  (X)')
plt.ylabel('GDP per capita(in $)  (Y)')
```

Out[21]:
```
Text(0, 0.5, 'GDP per capita(in $)  (Y)')
```



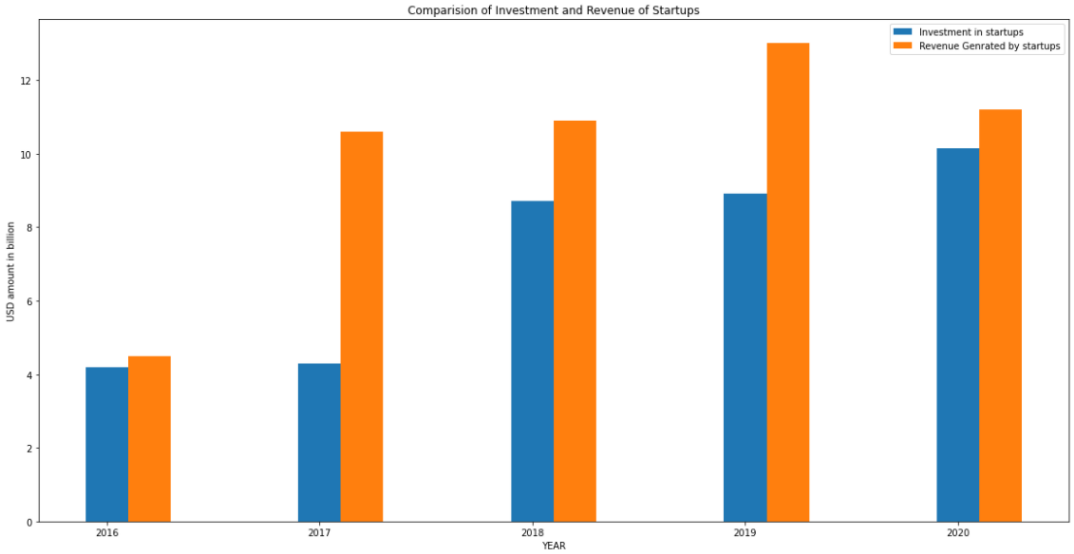
Objective 2.3

In [22]:
```
ps3=pd.read_csv("ps3.csv")
ps3
```

Out[22]:

	YEAR	Recognized Starts-ups	Investment in startups(USD amount in billion)	Revenue Genrated by startups (in USD billion)
0	2016	504	4.20	4.5
1	2017	5411	4.30	10.6
2	2018	8944	8.70	10.9
3	2019	11688	8.90	13.0
4	2020	14770	10.14	11.2

In [25]:
```
fig=plt.figure(figsize=(20,10))
width = 0.20
plt.bar(ps3['YEAR'],ps3['Investment in startups(USD amount in billion)'], width=width)
plt.bar(ps3['YEAR']+ width,ps3['Revenue Genrated by startups (in USD billion)'] , width=width)
plt.xticks(ticks=ps3['YEAR'], labels=ps3['YEAR'])
plt.xlabel("YEAR")
plt.ylabel("USD amount in billion")
plt.title("Comparision of Investment and Revenue of Startups")
plt.legend(("Investment in startups","Revenue Genrated by startups"))
plt.show()
```



In [23]:
```
from scipy import stats
t,a=stats.ttest_rel(ps3['Investment in startups(USD amount in billion)']
                    ,ps3['Revenue Genrated by startups (in USD billion)'])
t
```

Out[23]: -2.5714824295484084

In [24]:
```
talpha=2.132
if t>=talpha:
    print("accept null hypothesis")
else:
    print("reject null hypothesis")
```

```
reject null hypothesis
```

Appendix 3: Objective 3.1

In [19]:
```
# Import scipy.stats
import scipy.stats

# Two lists x and y
x = [289151.4,3881.8,28309.8,514537,71047.6,16214.6,886022.4,291112.6,31649.8,134822,504166.6,146813.2,585720.6,1627488.2,47007.
y = [49577103,1383727,31205576,104099452,25545198,1458545,60439692,25351462,6864602,32988134,61095297,33406061,72626809,11237433
print("Spearman's correlation:",scipy.stats.spearmanr(x, y)[0])
```

```
Spearman's correlation: 0.8975164499199446
```

```
In [20]: # Import pandas and scipy.stats
         mport pandas as pd
         mport scipy.stats

         # Two lists x and y
         : = [289151.4,3881.8,28309.8,514537,71047.6,16214.6,886022.4,291112.6,31649.8,134822,504166.6,146813.2,585720.6,1627488.2,47007.2,
         ' = [49577103,1383727,31205576,104099452,25545198,1458545,60439692,25351462,6864602,32988134,61095297,33406061,72626809,112374333,

         # Create a function that takes in x's and y's
         lef spearmans_rank_correlation(x, y):

             # Calculate the rank of x's
             xranks = pd.Series(x).rank()
             print("Rankings of X:")
             print(xranks)

             # Calculate the ranking of the y's
             yranks = pd.Series(y).rank()
             print("Rankings of Y:")
             print(yranks)

             # Calculate Pearson's correlation coefficient on the ranked versions of the data
             print("Spearman's Rank correlation:",scipy.stats.pearsonr(xranks, yranks)[0])

         # Call the function
         pearmans_rank_correlation(x, y)
```

```
Rankings of X:
0     25.0
1      6.0
2     16.0
3     32.0
4     20.0
5     14.0
6     35.0
7     26.0
8     17.0
9     21.0
10    31.0
11    23.0
12    33.0
13    37.0
14    18.0
15     5.0
16     7.0
17     4.0
18    22.0
19    24.0
20    30.0
21     3.0
22    36.0
23    29.0
24     9.0
25    34.0
26    19.0
27    27.0
28     8.0
29    15.0
30    11.0
31    12.0
```

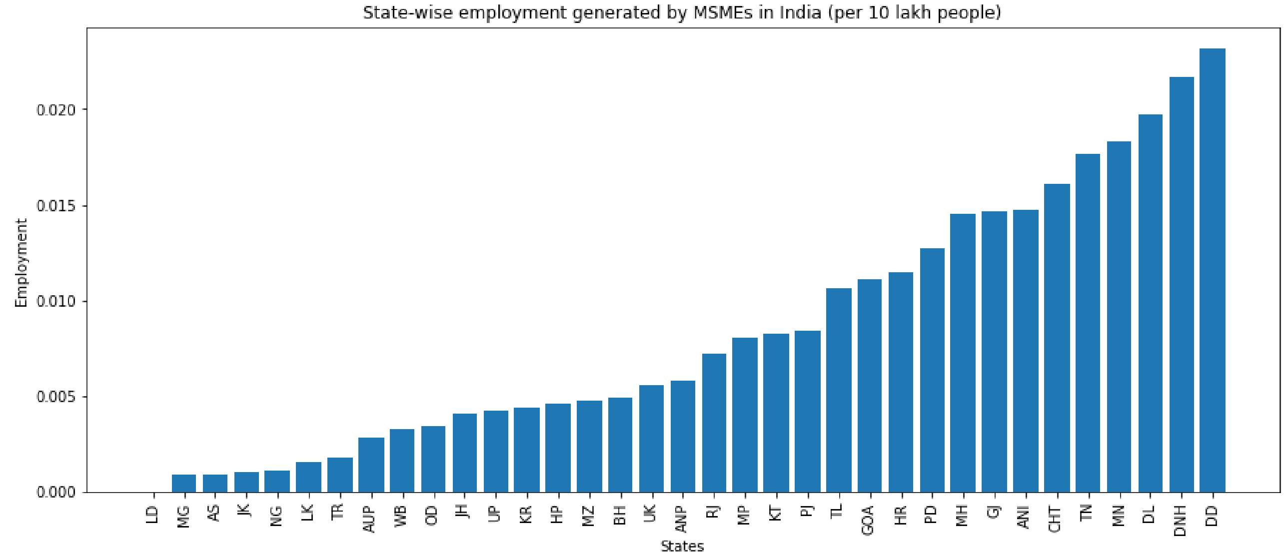
```
32    28.0
33    10.0
34     1.0
35     2.0
36    13.0
dtype: float64
Rankings of Y:
0     28.0
1     10.0
2     23.0
3     35.0
4     21.0
5     11.0
6     29.0
7     20.0
8     16.0
9     24.0
10    30.0
11    25.0
12    33.0
13    36.0
14    13.0
15    14.0
16     8.0
17    12.0
18    27.0
19    22.0
20    31.0
21     6.0
22    32.0
23    26.0
24    15.0
25    37.0
```

```
13    36.0
14    13.0
15    14.0
16     8.0
17    12.0
18    27.0
19    22.0
20    31.0
21     6.0
22    32.0
23    26.0
24    15.0
25    37.0
26    17.0
27    34.0
28     3.0
29     7.0
30     4.5
31     4.5
32    19.0
33    18.0
34     2.0
35     1.0
36     9.0
dtype: float64
Spearman's Rank correlation: 0.8975164499199446
```

Out[25]:

	Rank of x	States
0	1.0	LADAKH
1	2.0	LAKSHADWEEP
2	3.0	SIKKIM
3	4.0	NAGALAND
4	5.0	MEGHALAYA
5	6.0	ARUNACHAL PRADESH
6	7.0	MIZORAM
7	8.0	ANDAMAN AND NICOBAR ISLANDS
8	9.0	TRIPURA
9	10.0	JAMMU AND KASHMIR
10	11.0	DADAR AND NAGAR HAVELI
11	12.0	DAMAN AND DIU
12	13.0	PUDUCHERRY
13	14.0	GOA
14	15.0	CHANDIGARH
15	16.0	ASSAM
16	17.0	HIMACHAL PRADESH
17	18.0	MANIPUR
18	19.0	UTTARAKHAND
19	20.0	CHHATTISGARH
20	21.0	JHARKHAND
21	22.0	ODISHA
22	23.0	KERALA
23	24.0	PUNJAB
24	25.0	ANDHRA PRADESH
25	26.0	HARYANA
26	27.0	WEST BENGAL
27	28.0	DELHI
28	29.0	TELANGANA
29	30.0	RAJASTHAN
30	31.0	KARNATAKA
31	32.0	BIHAR
32	33.0	MADHYA PRADESH
33	34.0	UTTAR PRADESH
34	35.0	GUJARAT
35	36.0	TAMIL NADU
36	37.0	MAHARASHTRA
37	NaN	NaN

```python
In [1]: import pandas as pd
        ps1 = pd.read_csv("ps1 GRAPH.csv")
        ps1
        ps1_sort = ps1.sort_values("Rank")
        ps1_sort
        import matplotlib.pyplot as plt
        plt.figure(figsize=(15,6))
        plt.bar(ps1_sort["State"],ps1_sort["Rank"])
        plt.title("State-wise employment generated by MSMEs in India (per 10 lakh people)")
        plt.xticks(rotation = 90)
        plt.xlabel("States")
        plt.ylabel("Employment")
        plt.show()
```



Appendix 3: Objective 3.2

Out[15]:

	S No	state	2015	2019
0	1	ANP	1.97	2.78
1	2	BH	19.94	2.73
2	3	CHT	0.96	1.20
3	4	GOA	0.16	0.12
4	5	GJ	10.51	7.38
5	6	HR	0.91	2.58
6	7	HP	0.19	0.22
7	8	JH	4.23	1.25
8	9	KT	2.86	4.42
9	10	KR	2.29	1.40
10	11	MP	7.88	12.01
11	12	MH	10.92	27.02
12	13	MN	0.35	0.42
13	14	OD	1.73	1.05
14	15	PJ	0.95	3.41
15	16	RJ	6.84	6.89
16	17	SK	0.01	0.01
17	18	TN	8.41	10.03
18	19	TL	4.14	2.98
19	20	UP	9.10	6.32
20	21	UK	0.36	0.71
21	22	WB	3.31	1.62
22	23	CHT	0.06	0.17
23	24	DNH	0.05	0.06
24	25	DD	0.04	0.04
25	26	DL	1.53	2.33
26	27	PD	0.06	0.13

In [18]:
```
import pandas as pd
data = pd.read_csv("PS2 GRAPH.csv")
data

data.drop(columns=['state' , 'S No'], inplace = True)
data

data = pd.read_csv("PS2 GRAPH 1.csv")
data[['2015','2019']].describe()
```

Out[20]:

	2015	2019
count	27.000000	27.000000
mean	3.694815	3.677037
std	4.774684	5.645077
min	0.010000	0.010000
25%	0.270000	0.320000
50%	1.730000	1.620000
75%	5.535000	3.915000
max	19.940000	27.020000

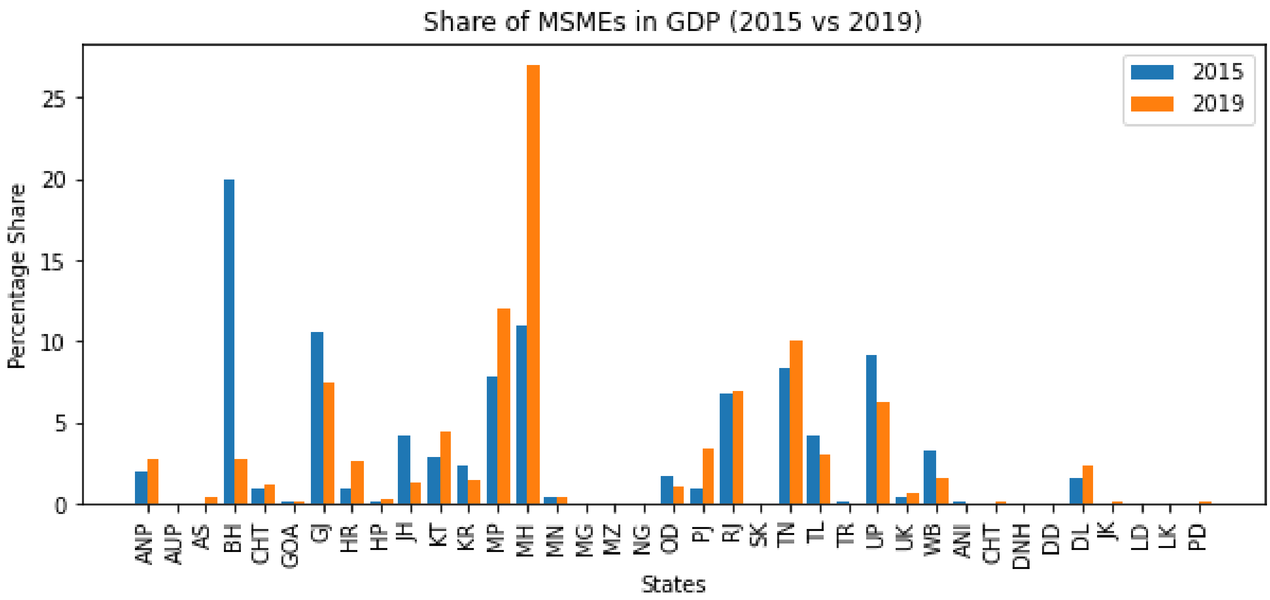
In [21]:
```
from scipy import stats
stats.shapiro(data['2015'])
stats.shapiro(data['2019'])
stats.ttest_rel(data['2015'], data['2019'])
```

Out[21]: Ttest_relResult(statistic=0.018921395145734484, pvalue=0.9850482706871498)

23]:
```
import pandas as pd
from scipy import stats

data[['2015','2019']].describe()
ttest,pval = stats.ttest_rel(data['2015'], data['2019'])
print(pval)

if pval<0.05:
    print("reject null hypothesis")
else:
    print("accept null hypothesis")
```

```
0.9850482706871498
accept null hypothesis
```

In [2]:
```
import pandas as pd
ps2 = pd.read_csv("ps2 GRAPH.csv")
ps2
import numpy as np
import matplotlib.pyplot as plt
years = ps2['State']
registered = ps2['2015']
employed = ps2['2019']
X_axis = np.arange(len(years))
plt.figure(figsize=(10,4))
plt.bar(X_axis - 0.2, registered, 0.4, label = '2015')
plt.bar(X_axis + 0.2, employed, 0.4, label = '2019')
plt.xticks(X_axis, years, rotation = 90)
plt.xlabel("States")
plt.ylabel("Percentage Share")
plt.title("Share of MSMEs in GDP (2015 vs 2019)")
plt.legend()
plt.show()
```



Appendix 3: Objective 3.3

```
In [3]: data = pd.read_csv("ps 3 graph.csv")
        data
```

	state	Manufacturing	Services
0	ANP	22271.0	40964.4
1	AUP	172.4	107.8
2	AS	1499.2	1250.0
3	BH	59633.4	115014.8
4	CHT	3680.4	7731.8
5	GOA	662.4	819.4
6	GJ	84364.2	65497.0
7	HR	14633.4	16755.2
8	HP	1212.0	1272.2
9	JH	9491.6	18657.0
10	KT	27308.2	29823.6
11	KR	12618.4	9173.4
12	MP	59282.6	111798.6
13	MH	100806.2	204413.4
14	MN	3593.0	2660.8
15	MG	272.8	127.2
16	MZ	357.2	202.6
17	NG	144.6	95.8
18	OD	7906.2	12378.2
19	PJ	15259.4	20341.6
20	RJ	42830.8	59804.8
21	SK	49.8	103.2
22	TN	87899.0	101479.0
23	TL	18680.6	35132.4
24	TR	533.6	499.2
25	UP	75924.6	85312.4
26	UK	3245.6	3710.2
27	WB	19928.8	19251.8
28	ANI	361.4	776.4
29	CHT	666.4	1267.0
30	DNH	442.4	414.2
31	DD	315.4	152.4
32	DL	12524.0	15067.8
33	JK	515.4	1072.6
34	LD	9.8	8.4
35	LK	824.6	1032.0

```
ps1['Y'] = ps1['Total']
ps1['X'] = range(1,len(ps1['Total'])+1)
ps1['XY'] = ps1['Y'] * ps1['X']
ps1['X^2'] = ps1['X'] * ps1['X']
ps1
```

	state	X	Y	XY	X^2
0	ANP	22271.0	40964.4	9.123182e+08	4.959974e+08
1	AUP	172.4	107.8	1.858472e+04	2.972176e+04
2	AS	1499.2	1250.0	1.874000e+06	2.247601e+06
3	BH	59633.4	115014.8	6.858724e+09	3.556142e+09
4	CHT	3680.4	7731.8	2.845612e+07	1.354534e+07
5	GOA	662.4	819.4	5.427706e+05	4.387738e+05
6	GJ	84364.2	65497.0	5.525602e+09	7.117318e+09
7	HR	14633.4	16755.2	2.451855e+08	2.141364e+08
8	HP	1212.0	1272.2	1.541906e+06	1.468944e+06
9	JH	9491.6	18657.0	1.770848e+08	9.009047e+07
10	KT	27308.2	29823.6	8.144288e+08	7.457378e+08
11	KR	12618.4	9173.4	1.157536e+08	1.592240e+08
12	MP	59282.6	111798.6	6.627712e+09	3.514427e+09
13	MH	100806.2	204413.4	2.060614e+10	1.016189e+10
14	MN	3593.0	2660.8	9.560254e+06	1.290965e+07
15	MG	272.8	127.2	3.470016e+04	7.441984e+04
16	MZ	357.2	202.6	7.236872e+04	1.275918e+05
17	NG	144.6	95.8	1.385268e+04	2.090916e+04
18	OD	7906.2	12378.2	9.786452e+07	6.250800e+07
19	PJ	15259.4	20341.6	3.104006e+08	2.328493e+08
20	RJ	42830.8	59804.8	2.561487e+09	1.834477e+09
21	SK	49.8	103.2	5.139360e+03	2.480040e+03
22	TN	87899.0	101479.0	8.919903e+09	7.726234e+09
23	TL	18680.6	35132.4	6.562943e+08	3.489648e+08
24	TR	533.6	499.2	2.663731e+05	2.847290e+05
25	UP	75924.6	85312.4	6.477310e+09	5.764545e+09
26	UK	3245.6	3710.2	1.204183e+07	1.053392e+07
27	WB	19928.8	19251.8	3.836653e+08	3.971571e+08
28	ANI	361.4	776.4	2.805910e+05	1.306100e+05
29	CHT	666.4	1267.0	8.443288e+05	4.440890e+05
30	DNH	442.4	414.2	1.832421e+05	1.957178e+05
31	DD	315.4	152.4	4.806696e+04	9.947716e+04
32	DL	12524.0	15067.8	1.887091e+08	1.568506e+08
33	JK	515.4	1072.6	5.528180e+05	2.656372e+05
34	LD	9.8	8.4	8.232000e+01	9.604000e+01
35	LK	824.6	1032.0	8.509872e+05	6.799652e+05
36	PD	689920.8	984168.6	6.789984e+08	4.759907e+07

```
: import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns

data = pd.read_csv("ps 3 graph.csv")
data

data.drop(columns=['state'], inplace = True)
data

x = data[['Manufacturing']]
y = data['Services']

from sklearn.model_selection import train_test_split
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size = 0.3, random_state = 100)
2
from sklearn.linear_model import LinearRegression
slr = LinearRegression()
slr.fit(x_train, y_train)

print("Intercept: ", slr.intercept_)
print("Coefficient: ", slr.coef_)

#Prediction of test set
y_pred_slr= slr.predict(x_test)
#Predicted values
print("Prediction for test set: {}".format(y_pred_slr))
```

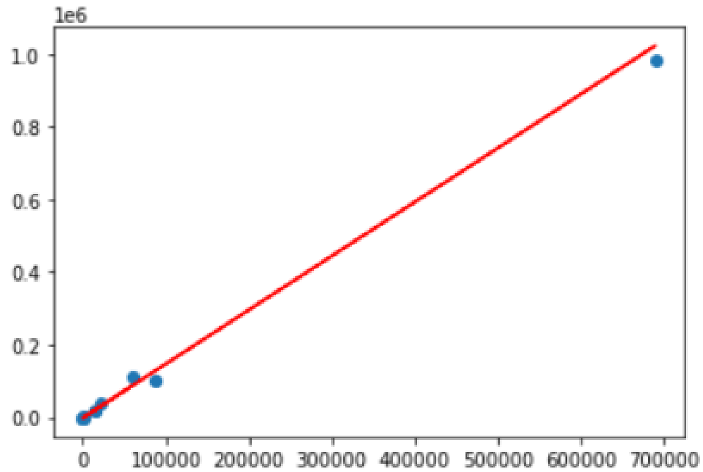
```
slr_diff = pd.DataFrame({'Actual value': y_test, 'Predicted value': y_pred_slr})
slr_diff


from sklearn import metrics
meanAbErr = metrics.mean_absolute_error(y_test, y_pred_slr)
meanSqErr = metrics.mean_squared_error(y_test, y_pred_slr)
rootMeanSqErr = np.sqrt(metrics.mean_squared_error(y_test, y_pred_slr))
print('R squared: {:.2f}'.format(slr.score(x,y)*100))
print('Mean Absolute Error:', meanAbErr)
print('Mean Square Error:', meanSqErr)
print('Root Mean Square Error:', rootMeanSqErr)
```

```
Intercept:  -1413.723208261501
Coefficient:  [1.48529713]
Prediction for test set: [ 3.16653293e+04 -4.23921198e+02 -7.56627756e+02 -1.88947192e+02
  1.29142410e+05  1.02332366e+06  2.03212239e+04  8.66385527e+04
 -1.39916730e+03 -4.29862387e+02 -1.15765798e+03  2.12510199e+04]
R squared: 98.81
Mean Absolute Error: 9479.83491197433
Mean Square Error: 253524297.08379373
Root Mean Square Error: 15922.44632849468
```

```
In [ ]: import pandas as pd
ps3 = pd.read_csv("ps3 GRAPH.csv")
ps3
import numpy as np
import matplotlib.pyplot as plt
years = ps3['State']
registered = ps3['Manufacturing']
employed = ps3['Services']
X_axis = np.arange(len(years))
plt.figure(figsize=(15,6))
plt.bar(X_axis - 0.2, registered, 0.4, label = 'Manufacturing')
plt.bar(X_axis + 0.2, employed, 0.4, label = 'Services')
plt.xticks(X_axis, years, rotation = 90)
plt.xlabel("States")
plt.ylabel("Number of registrations")
plt.title("Number of registrations of manufacturing enterprises vs services enterprises")
plt.legend()
plt.show()
```


```
# regression equation: services = -1413.723 + 1.48*manufacturing
```

```
plt.scatter(x_test,y_test)
plt.plot(x_test, y_pred_slr, 'Red')
plt.show()
```



Appendix 4: Objective 4.1

```
In [6]: import numpy as np
        import pandas as pd
        import matplotlib.pyplot as plt
```

```
In [7]: dt = pd.read_csv("PS1_How tourism is affecting the employment in India.csv")
        dt
```

Out[7]:

	Years	Total Employees in the tourism sector (in Millions)	Foreign Tourist Arrivals (in Millions)
0	2014	67.20	7.8
1	2015	69.75	8.1
2	2016	72.26	8.8
3	2017	75.34	10.2
4	2018	80.63	10.6
5	2019	87.50	10.9

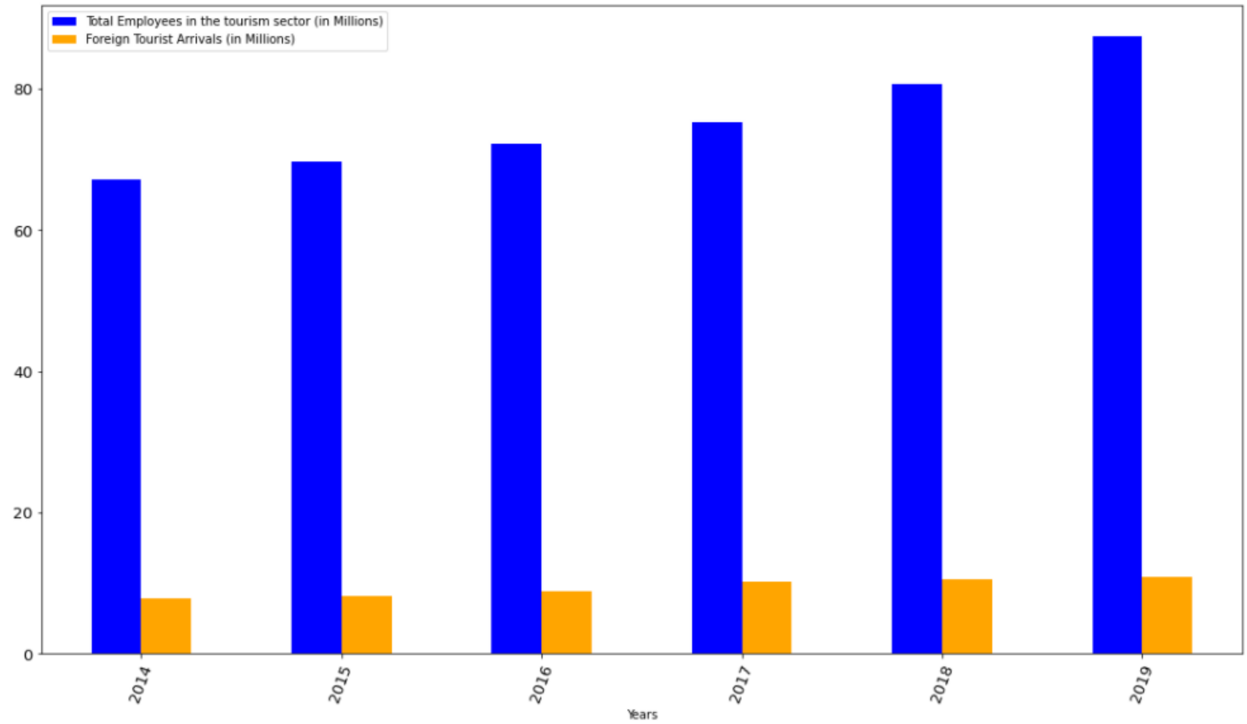
```
In [4]: dt.describe()
```

Out[4]:

	Years	Total Employees in the tourism sector (in Millions)	Foreign Tourist Arrivals (in Millions)
count	6.000000	6.000000	6.000000
mean	2016.500000	75.446667	9.400000
std	1.870829	7.520313	1.337161
min	2014.000000	67.200000	7.800000
25%	2015.250000	70.377500	8.275000
50%	2016.500000	73.800000	9.500000
75%	2017.750000	79.307500	10.500000
max	2019.000000	87.500000	10.900000

```
In [5]: j = dt[['Years','Total Employees in the tourism sector (in Millions)','Foreign Tourist Arrivals (in Millions)']]
```

```
In [6]: j.plot.bar(x="Years",figsize=(18,10),fontsize=13,rot=70,color=("blue","orange"))
Out[6]: <AxesSubplot:xlabel='Years'>
```



Appendix 4: Objective 4.2

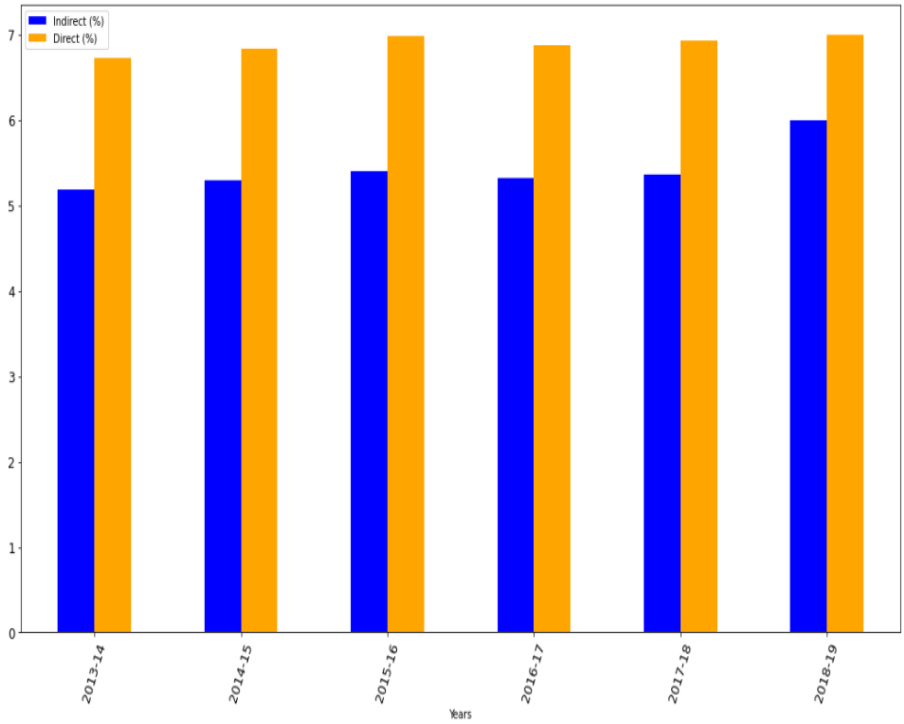
In [2]:
```
dt1 = pd.read_csv("DIRECT AND INDIRECT EMPLOYMENT BY TOURISM.csv")
dt1
```

Out[2]:

	Years	Shares in Jobs (%)	Indirect (%)	Direct (%)	Direct+Indirect (in Millions)
0	2013-14	11.91	5.19	6.72	67.19
1	2014-15	12.14	5.30	6.84	69.56
2	2015-16	12.38	5.40	6.98	72.26
3	2016-17	12.20	5.32	6.88	75.71
4	2017-18	12.29	5.36	6.93	80.54
5	2018-19	13.00	6.00	7.00	89.00

In [8]:
```
k = dt1[['Years','Indirect (%)','Direct (%)']]
k.plot.bar(x="Years",figsize=(18,10),fontsize=13,rot=70,color=("blue","orange"))
```

Out[8]: <AxesSubplot:xlabel='Years'>



In [9]:
```
dt1.describe()
```

Out[9]:

	Shares in Jobs (%)	Indirect (%)	Direct (%)	Direct+Indirect (in Millions)
count	6.00000	6.000000	6.000000	6.000000
mean	12.32000	5.428333	6.891667	75.710000
std	0.36927	0.288889	0.103231	8.028305
min	11.91000	5.190000	6.720000	67.190000
25%	12.15500	5.305000	6.850000	70.235000
50%	12.24500	5.340000	6.905000	73.985000
75%	12.35750	5.390000	6.967500	79.332500
max	13.00000	6.000000	7.000000	89.000000

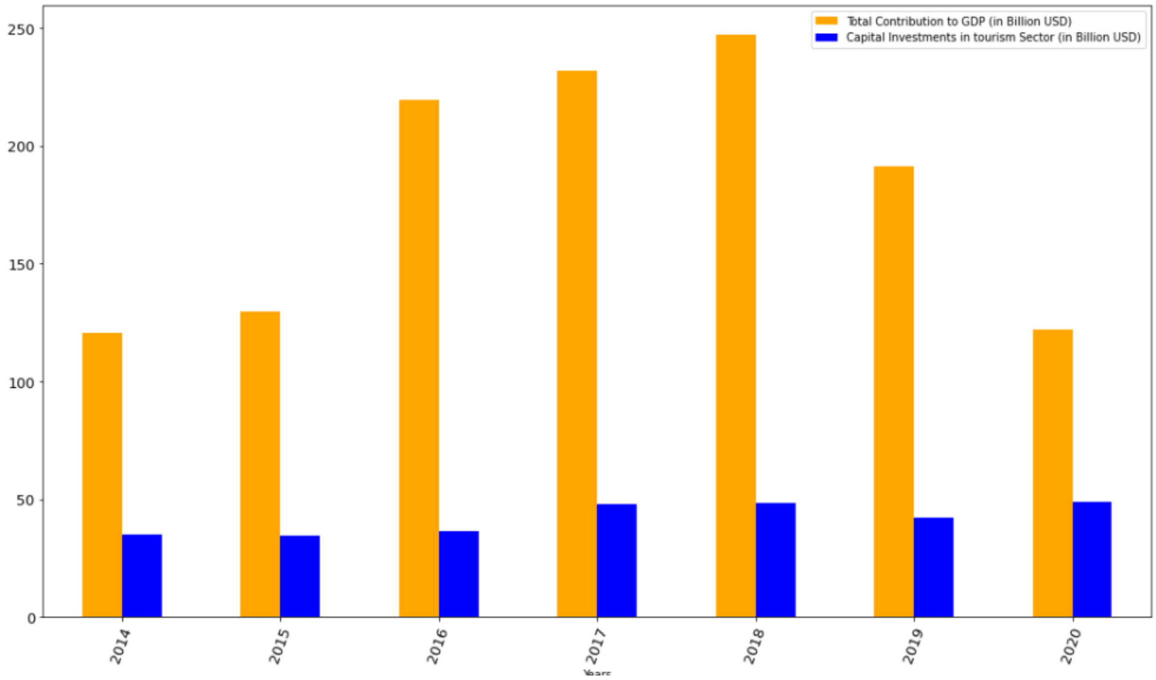
In [10]:
```
dt2 = pd.read_csv("Contribution in the GDP ny the tourism Industry.csv")
dt2
```

Out[10]:

	Years	Total Contribution to GDP (in Billion USD)	Capital Investments in tourism Sector (in Billion USD)
0	2014	120.6	35.0
1	2015	129.5	34.6
2	2016	219.7	36.6
3	2017	232.0	47.8
4	2018	247.4	48.4
5	2019	191.3	42.3
6	2020	121.9	49.0

```
In [11]: l = dt2[['Years','Total Contribution to GDP (in Billion USD)','Capital Investments in tourism Sector (in Billion USD)']]
```

```
In [12]: l.plot.bar(x="Years",figsize=(18,10),fontsize=13,rot=70,color=("orange","blue"))

Out[12]: <AxesSubplot:xlabel='Years'>
```




```
In [8]: data = dt[['Total Employees in the tourism sector (in Millions)','Foreign Tourist Arrivals (in Millions)']]
        correlation = data.corr()
        print(correlation)

                                                    Total Employees in the tourism sector (in Millions)  \
Total Employees in the tourism sector (in Milli...                                           1.000000
Foreign Tourist Arrivals (in Millions)                                                       0.929348

                                                    Foreign Tourist Arrivals (in Millions)
Total Employees in the tourism sector (in Milli...                                0.929348
Foreign Tourist Arrivals (in Millions)                                            1.000000
```

In [9]:
```
def correlation(r):
    r=float(r)
    if r>-0.1 and R_value<r:
        print("No Correlation")
    elif r>=0.1 and r<0.5:
        print("Weakly Positive Correlation")
    elif r>=0.5 and r<0.8:
        print("Moderately Positive Correlation")
    elif r>=0.8 and r<1.0:
        print("Strongly Positive Correlation")
    elif r>=-0.1 and r<-0.5:
        print("Weekly Negative Correlation")
    elif r>=-0.5 and r<-0.8:
        print("Moderate Negative Correlation")
    elif r>=-0.8 and r<-1.0:
        print("Moderate Negative Correlation")
```

```
data2 = dt2[['Total Contribution to GDP (in Billion USD)','Capital Investments in tourism Sector (in Billion USD)']]
correlation2 = data2.corr()
print(correlation2)
```

```
                                              Total Contribution to GDP (in Billion USD)  \
Total Contribution to GDP (in Billion USD)                                      1.000000
Capital Investments in tourism Sector (in Billi...                              0.416396

                                              Capital Investments in tourism Sector (in Billion USD)
Total Contribution to GDP (in Billion USD)                                               0.416396
Capital Investments in tourism Sector (in Billi...                                       1.000000
```

:
```
data1 = dt1[['Direct (%)','Indirect (%)']]
correlation1 = data1.corr()
print(correlation1)
```

```
              Direct (%)  Indirect (%)
Direct (%)      1.000000      0.708307
Indirect (%)    0.708307      1.000000
```

```
In [16]: dt1.describe()
```

Out[16]:

	Shares in Jobs (%)	Indirect (%)	Direct (%)	Direct+Indirect (in Millions)
count	6.00000	6.000000	6.000000	6.000000
mean	12.32000	5.428333	6.891667	75.710000
std	0.36927	0.288889	0.103231	8.028305
min	11.91000	5.190000	6.720000	67.190000
25%	12.15500	5.305000	6.850000	70.235000
50%	12.24500	5.340000	6.905000	73.985000
75%	12.35750	5.390000	6.967500	79.332500
max	13.00000	6.000000	7.000000	89.000000

```
import pandas as pd
from scipy import stats

dt1[['Direct (%)','Indirect (%)']].describe()
ttest,pval = stats.ttest_rel(dt1['Direct (%)'], dt1['Indirect (%)'])
print(pval)

if pval<0.05:
    print("reject null hypothesis")
else:
    print("accept null hypothesis")
```

```
1.8828987995349395e-05
reject null hypothesis
```

ABBREVIATIONS

AN	Andaman and Nicobar
AP	Andhra Pradesh
AR	Arunachal Pradesh
AS	Assam
BH	Bihar
CG	Chhattisgarh
GA	Goa
GJ	Gujarat
HR	Haryana
HP	Himanchal Pradesh
JK	Jammu and Kashmir
JH	Jharkhand
KR	Karnataka
KL	Kerala
LD	Lakshadweep
MP	Madhya Pradesh
MH	Maharashtra
MN	Manipur
MG	Meghalaya
MZ	Mizoram
NL	Nagaland
OD	Odisha
PD	Puducherry
PB	Punjab
RJ	Rajasthan
SK	Sikkim
TN	Tamil Nadu
TR	Tripura
UP	Uttar Pradesh
UK	Uttarakhand
WB	West Bengal

INDIVIDUAL CONTRIBUTION

Student ID	Name	Objective(s)	Summary on Contribution (Brief method, result summary and conclusion)
2021BTech012	Anay Sinhal	To Analyse Employment Generation by Government Policy (MGNREGA)	Literature Review List of Tables and Figures Methodologies and Graphs Document Formatting
2021BTech031	Babar Khilji	To analyse how start-ups affect economy, employment, and its different aspects.	Abstract, Introduction, Literature review, methodologies, and graphs
2021BTech041	Diya Mathur	To analyse how MSMEs affect Economy and Employment.	Literature Review Methodologies and Graphs
2021BTech058	Kounen Aftab	To study employment opportunities created by the tourism industries.	Literature Review, Introduction, List of Figures and Graphs