In [1]:

```
# Carga de librerias
import pandas as pd
import numpy as np

import warnings
warnings.filterwarnings(action='ignore')

import matplotlib.pyplot as plt
import seaborn as sns
```

In [2]:

```
# Lectura de los datos
df = pd.read_csv("solicitud_creditos_info.csv")
print(df.info())
df.head()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 251617 entries, 0 to 251616
Data columns (total 33 columns):
 #   Column                    Non-Null Count   Dtype  
---  ------                    --------------   -----  
 0   index                     251617 non-null  int64  
 1   default                   251617 non-null  int64  
 2   contract_type             251617 non-null  object 
 3   gender                    251617 non-null  object 
 4   flag_vehicle              251617 non-null  object 
 5   flag_properties           251617 non-null  object 
 6   num_children              251617 non-null  int64  
 7   total_income              251617 non-null  float64
 8   loan_amount               251617 non-null  float64
 9   income_type               251617 non-null  object 
 10  marital_status            251617 non-null  object 
 11  age                       251617 non-null  float64
 12  work_age                  251617 non-null  float64
 13  flag_cellphone            251617 non-null  int64  
 14  flag_work_phone           251617 non-null  int64  
 15  flag_work_company         251617 non-null  int64  
 16  flag_cellphone_answer     251617 non-null  int64  
 17  flag_homephone            251617 non-null  int64  
 18  flag_email                251617 non-null  int64  
 19  type_organisation         251617 non-null  object 
 20  external_score_1          251617 non-null  float64
 21  external_score_2          251617 non-null  float64
 22  age_mobilephone_days      251617 non-null  float64
 23  num_petic_bureau_day      251617 non-null  float64
 24  num_petic_bureau_week     251617 non-null  float64
 25  num_petic_bureau_month    251617 non-null  float64
 26  num_petic_bureau_quarter  251617 non-null  float64
 27  num_petic_bureau_year     251617 non-null  float64
 28  block_month               251617 non-null  int64  
 29  year                      251617 non-null  int64  
 30  month                     251617 non-null  int64  
 31  Office                    251617 non-null  int64  
 32  Employee                  251617 non-null  int64  
dtypes: float64(12), int64(14), object(7)
memory usage: 63.3+ MB
None
```

Out[2]:

	index	default	contract_type	gender	flag_vehicle	flag_properties	num_children	total_income	loan_amount
0	0	1	personal loan	M	N	Y	0	394338.0	1207961.0
1	1	1	personal loan	F	N	Y	0	322796.0	409690.0

	index	default	contract_type	gender	flag_vehicle	flag_properties	num_children	total_income	loan_amount
2	2	1	personal loan	M	N	Y	1	241201.0	858450.0
3	3	1	personal loan	M	N	N	1	264255.0	114679.0
4	4	1	personal loan	F	N	Y	1	166682.0	402669.0

5 rows × 33 columns

In [3]:
```
df['Office'] = df['Office'].astype(str)
```

Cálculo de diferentes KPI

In [4]:
```
# Columna con el mes en formato fecha
df['fecha'] = df['year'].astype(str) + '/' + df['month'].astype(str) + '/28'
df['fecha'] = pd.to_datetime(df['fecha'])
```

KPI de valor de créditos gestionados mensualmente por cada empleado

Esto permite hallar los gestores de crédito (empleados) con mejor y peor rendimiento

In [5]:
```
# Un KPI relevante podria ser el valor total de los créditos gestionados por los empleados
df_KPI_gestores = pd.pivot_table(df, index = ['Employee'], columns = 'fecha', values='loan
df_KPI_gestores
```

Out[5]:

fecha	2017-01-28	2017-02-28	2017-03-28	2017-04-28	2017-05-28	2017-06-28	2017-07-28	2017
Employee								
10	127472223.0	123547609.0	130449910.0	138033208.0	138726383.0	137055449.0	114786331.0	1287
11	57336076.0	45831262.0	38427385.0	58055557.0	61298505.0	46681385.0	46054460.0	492
13	4231591.0	2413467.0	3415021.0	1851734.0	2421760.0	2197719.0	1081050.0	17
14	169015987.0	155014532.0	162397332.0	152322262.0	148616959.0	170664952.0	142156244.0	1495
16	22692667.0	21849547.0	31440567.0	21894677.0	34573636.0	31040045.0	24484420.0	222
17	36730220.0	38696855.0	33097710.0	42885393.0	47826780.0	38731074.0	40013761.0	379
18	130174602.0	132451855.0	125629079.0	138602066.0	136860009.0	135671059.0	130866398.0	1089
27	37483755.0	36533348.0	28723306.0	39084734.0	22545668.0	28522713.0	28850909.0	323
29	256284523.0	239865661.0	225837610.0	289306435.0	247903738.0	246228619.0	258766634.0	256
31	240176135.0	269236405.0	247235408.0	310845707.0	273385627.0	285065214.0	247760532.0	2585
33	1520045.0	1395632.0	4567319.0	2559194.0	3805933.0	1092984.0	2894527.0	28
35	113558866.0	99648769.0	102067856.0	108865462.0	119096836.0	115203145.0	98470591.0	1097
36	61969809.0	69072212.0	64663446.0	70623920.0	76418979.0	59577687.0	60166263.0	620
37	11435386.0	8599981.0	12410147.0	11485476.0	15456499.0	14571054.0	18699433.0	121
38	240833601.0	218025226.0	252520005.0	262542396.0	272054138.0	241764930.0	239513121.0	2317
45	9789190.0	17461830.0	20865765.0	16790222.0	16341334.0	22095208.0	5812614.0	126
47	121250653.0	126428029.0	124989863.0	139331548.0	118718391.0	118302190.0	109125274.0	1248
48	34411975.0	38198724.0	52889366.0	39944329.0	49535626.0	37983575.0	47078127.0	415

fecha	2017-01-28	2017-02-28	2017-03-28	2017-04-28	2017-05-28	2017-06-28	2017-07-28	2017
Employee								
52	86020473.0	86926747.0	72785781.0	64838210.0	85143032.0	87798038.0	62481312.0	744
56	61705132.0	82632039.0	65845944.0	93512995.0	77313261.0	67200090.0	77539023.0	806
58	137590826.0	115340076.0	127680719.0	168075189.0	157371956.0	138416002.0	124150508.0	1573
67	242876982.0	258601964.0	265906363.0	309253378.0	289964787.0	266070215.0	253900822.0	2373
69	22966887.0	29781706.0	32771605.0	25632461.0	25500122.0	27767285.0	26746998.0	204
72	372302031.0	432952052.0	407454043.0	468654553.0	412170451.0	476456164.0	390042039.0	4058
76	272765143.0	217290833.0	271375941.0	272746520.0	275194635.0	248988455.0	227949603.0	248
78	193871974.0	208422770.0	220310180.0	220857420.0	228157785.0	203007243.0	219184603.0	1954
79	37350077.0	26418379.0	31471435.0	41680410.0	44945891.0	38849935.0	41737126.0	264
81	48283523.0	46497364.0	46149254.0	55501702.0	51436902.0	57910132.0	71701685.0	456
83	173910436.0	146051430.0	163961557.0	217739975.0	165961527.0	208238575.0	185923315.0	1650
86	504827645.0	510358530.0	505914989.0	575978038.0	561859543.0	580370144.0	502046344.0	472
87	205075762.0	192927436.0	197421857.0	221401849.0	202262767.0	214434821.0	191216675.0	1847
98	34544020.0	17562142.0	25841675.0	21794982.0	30270767.0	27322965.0	21935728.0	197

32 rows × 36 columns

Con esto nos puede interesar mostrar los empleados cuyas ventas son las más altas y más bajas de acuerdo al último mes

In [6]:
```
# Top 3 empleados con menor performance
df_KPI_gestores.sort_values('2019-12-28').head(3) # Ordenar de acuerdo al último mes y tor
```

Out[6]:

fecha	2017-01-28	2017-02-28	2017-03-28	2017-04-28	2017-05-28	2017-06-28	2017-07-28	2017-08-28	2017
Employee									
33	1520045.0	1395632.0	4567319.0	2559194.0	3805933.0	1092984.0	2894527.0	2879270.0	2089
13	4231591.0	2413467.0	3415021.0	1851734.0	2421760.0	2197719.0	1081050.0	1730012.0	2630
45	9789190.0	17461830.0	20865765.0	16790222.0	16341334.0	22095208.0	5812614.0	12672161.0	18003

3 rows × 36 columns

In [7]:
```
# Top 3 empleados con mejor performance
df_KPI_gestores.sort_values('2019-12-28').tail(3) # Ordenar de acuerdo al último mes y tor
```

Out[7]:

fecha	2017-01-28	2017-02-28	2017-03-28	2017-04-28	2017-05-28	2017-06-28	2017-07-28	2017
Employee								
67	242876982.0	258601964.0	265906363.0	309253378.0	289964787.0	266070215.0	253900822.0	2373
72	372302031.0	432952052.0	407454043.0	468654553.0	412170451.0	476456164.0	390042039.0	4058
86	504827645.0	510358530.0	505914989.0	575978038.0	561859543.0	580370144.0	502046344.0	4727

3 rows × 36 columns

In [8]:
```
# Grafiquemos
pd.concat([df_KPI_gestores.sort_values('2019-12-28').head(3),
           df_KPI_gestores.sort_values('2019-12-28').tail(3)]).T.plot()
```

Out[8]:
```
<AxesSubplot:xlabel='fecha'>
```

KPI de ratio de colocación de tarjetas de crédito por oficina

Se puede estar interesado en que la colocación de tarjetas de crédito se incremente por que son más rentables para la organización. Para ello el seguimiento de este KPI puede ser útil.

In [9]:
```
df_group_ofi_tipo = df.groupby(['Office', 'year', 'contract_type'])['loan_amount'].sum().r
df_group_ofi_tipo['total'] = df_group_ofi_tipo.groupby(['Office', 'year'])['loan_amount'].
df_group_ofi_tipo['ratio'] =  round(df_group_ofi_tipo['loan_amount'] * 100 / df_group_ofi_
df_group_ofi_tipo = df_group_ofi_tipo.query("contract_type=='credit card'")  # dejar solo

#  Datos
df_group_ofi_tipo
```

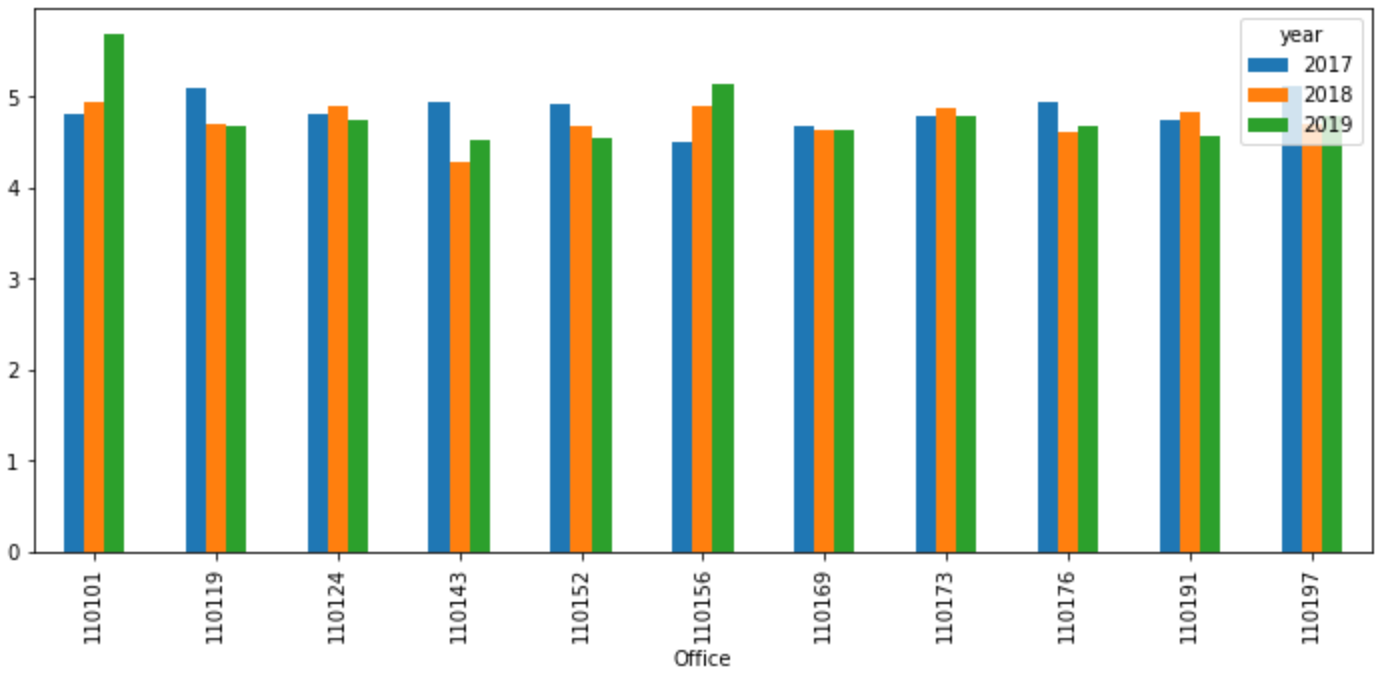
Out[9]:

	Office	year	contract_type	loan_amount	total	ratio
0	110101	2017	credit card	95415089.0	1.982337e+09	4.81
2	110101	2018	credit card	107352653.0	2.166881e+09	4.95
4	110101	2019	credit card	125104228.0	2.194785e+09	5.70
6	110119	2017	credit card	214704329.0	4.216363e+09	5.09
8	110119	2018	credit card	207950729.0	4.423001e+09	4.70
10	110119	2019	credit card	207613774.0	4.436347e+09	4.68
12	110124	2017	credit card	174318004.0	3.620665e+09	4.81
14	110124	2018	credit card	194961444.0	3.981137e+09	4.90
16	110124	2019	credit card	198371062.0	4.173058e+09	4.75
18	110143	2017	credit card	214777551.0	4.351053e+09	4.94
20	110143	2018	credit card	214722562.0	5.002745e+09	4.29
22	110143	2019	credit card	228761013.0	5.050047e+09	4.53
24	110152	2017	credit card	311871167.0	6.331170e+09	4.93

	Office	year	contract_type	loan_amount	total	ratio
26	110152	2018	credit card	323940950.0	6.923693e+09	4.68
28	110152	2019	credit card	315252725.0	6.950907e+09	4.54
30	110156	2017	credit card	178254714.0	3.955997e+09	4.51
32	110156	2018	credit card	217188098.0	4.426206e+09	4.91
34	110156	2019	credit card	226162306.0	4.398842e+09	5.14
36	110169	2017	credit card	209804342.0	4.472558e+09	4.69
38	110169	2018	credit card	230770390.0	4.978964e+09	4.63
40	110169	2019	credit card	229796947.0	4.956447e+09	4.64
42	110173	2017	credit card	178936050.0	3.742545e+09	4.78
44	110173	2018	credit card	195926059.0	4.014133e+09	4.88
46	110173	2019	credit card	189558505.0	3.951890e+09	4.80
48	110176	2017	credit card	291315799.0	5.893199e+09	4.94
50	110176	2018	credit card	292169803.0	6.318229e+09	4.62
52	110176	2019	credit card	297659351.0	6.369045e+09	4.67
54	110191	2017	credit card	372779958.0	7.865678e+09	4.74
56	110191	2018	credit card	420317181.0	8.680404e+09	4.84
58	110191	2019	credit card	398337492.0	8.734638e+09	4.56
60	110197	2017	credit card	253096006.0	4.937204e+09	5.13
62	110197	2018	credit card	256601418.0	5.456562e+09	4.70
64	110197	2019	credit card	264084185.0	5.513542e+09	4.79

In [10]:
```
# Gráfico
df_group_ofi_tipo = pd.pivot_table(df_group_ofi_tipo, index='Office', columns = 'year', va
df_group_ofi_tipo.plot(kind='bar', figsize=(12,5))
```

Out[10]:
```
<AxesSubplot:xlabel='Office'>
```



No hay una tendencia general, algunas oficinas incrementan el ratio y otras lo disminuyen. Resulta interesante preguntarnos por qué ocurre esto y explorarlo a través de los datos en búsqueda de una respuesta. Sin embargo, ¿tenemos la información suficiente para responder a esta pregunta?

Es posible crear muchos más KPIs con la información que tenemos, ya dependerá de las necesidades de la empresa, ydel entendimiento de las necesidades y de los procesos por parte del científico de datos e incluso de su creatividad e ingenio.

Asociación a través de un modelo lineal

Ayuda en la toma de decisiones de selección de personal

Ejemplo de un modelo que nos permitirá determinar los factores asociados a la productividad de los empleados

In [13]:
```
# Creación de la variable de productividad de los empleados (en este caso número de crédi
# Se pueden crear otras variables de productividad (valor de los créditos, evolución de l

df_target_e = df.query("year==2019").groupby('Employee').count()['index'].to_frame().reset
df_target_e.rename({'index':'Productividad'}, inplace=True, axis=1)  # Renombrar la colum
df_target_e.head()
```

Out[13]:

	Employee	Productividad
0	10	2491
1	11	991
2	13	78
3	14	3110
4	16	468

Para cada empleado se tiene una medida de productividad. Ahora agreguemos las caracteristicas observadas en el momento de la contratación e indaguemos sobre los factores relevantes en la productividad

In [14]:
```
df_employees = pd.read_csv("Employees.csv")
df_employees.info()
df_employees.head()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 32 entries, 0 to 31
Data columns (total 7 columns):
 #   Column               Non-Null Count  Dtype  
---  ------               --------------  -----  
 0   Employee             32 non-null     int64  
 1   employee_age         32 non-null     int64  
 2   employe_gender       32 non-null     object 
 3   exp_previa           32 non-null     int64  
 4   employee_estudios    32 non-null     object 
 5   resul_prub_seleccion 32 non-null     float64
 6   eval_jefeRH          32 non-null     float64
dtypes: float64(2), int64(3), object(2)
memory usage: 1.9+ KB
```

Out[14]:

	Employee	employee_age	employe_gender	exp_previa	employee_estudios	resul_prub_seleccion	eval_jefeRH
0	10	38	F	10	Secud. Completa	8.6	8.4
1	11	35	M	4	Técnico	8.4	8.7

	Employee	employee_age	employe_gender	exp_previa	employee_estudios	resul_prub_seleccion	eval_jefeRH
2	13	24	M	1	Uni. Completo	9.2	9.1
3	14	44	F	12	Secud. Completa	8.1	8.9
4	16	32	M	1	Uni. Completo	8.3	8.4

In [15]:
```
df_target_e = pd.merge(df_target_e, df_employees, on = 'Employee') # Unir los datos de ca
df_target_e.head()
```

Out[15]:

	Employee	Productividad	employee_age	employe_gender	exp_previa	employee_estudios	resul_prub_seleccion
0	10	2491	38	F	10	Secud. Completa	8.6
1	11	991	35	M	4	Técnico	8.4
2	13	78	24	M	1	Uni. Completo	9.2
3	14	3110	44	F	12	Secud. Completa	8.1
4	16	468	32	M	1	Uni. Completo	8.3

In [16]:
```
# Exporto estos datos (pueden ser utiles despues)
df_target_e.to_csv('Employee_target.csv', index=False)
```

In [17]:
```
# Librerias para modelos estadísticos
import statsmodels.api as sm
```

In [18]:
```
y = np.log(df_target_e['Productividad']) # Variable objetivo en el objeto y es el logaritr

X = df_target_e[['exp_previa', 'resul_prub_seleccion', 'eval_jefeRH', 'employee_estudios']
X = pd.get_dummies(X, drop_first=True) # Convertir valores categóricas en numericas
X['Constante'] = 1 # Termino constante
```

In [19]:
```
mod = sm.OLS(y, X) # Especificar el modelo
res = mod.fit() # Ajustar el modelo
print(res.summary()) # Resulmen del modelo en pantalla
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:          Productividad   R-squared:                       0.815
Model:                            OLS   Adj. R-squared:                  0.770
Method:                 Least Squares   F-statistic:                     18.30
Date:                Sun, 02 Oct 2022   Prob (F-statistic):           4.80e-08
Time:                        19:19:05   Log-Likelihood:                -25.972
No. Observations:                  32   AIC:                             65.94
Df Residuals:                      25   BIC:                             76.20
Df Model:                           6
Covariance Type:            nonrobust
================================================================================================
                                   coef    std err          t      P>|t|      [0.025
     0.975]
------------------------------------------------------------------------------------------------
exp_previa                       0.1948      0.023      8.595      0.000       0.148
      0.242
resul_prub_seleccion            -0.0133      0.218     -0.061      0.952      -0.462
      0.436
```

```
eval_jefeRH                          0.1341      0.268      0.500      0.621      -0.418
      0.687
employee_estudios_Técnico            0.0467      0.246      0.190      0.851      -0.461
      0.554
employee_estudios_Uni. Completo     -0.3980      0.353     -1.126      0.271      -1.126
      0.330
employee_estudios_Uni. Incompleto   -1.3890      0.494     -2.814      0.009      -2.405
     -0.372
Constante                            4.8652      2.987      1.629      0.116      -1.287
     11.018
==============================================================================
Omnibus:                        1.433   Durbin-Watson:                   1.909
Prob(Omnibus):                  0.488   Jarque-Bera (JB):                0.509
Skew:                          -0.200   Prob(JB):                        0.775
Kurtosis:                       3.471   Cond. No.                         411.
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specifie
d.
```

Considerando un nivel de significancia del 5% (0.05) y controlando por otros factores la experiencia previa es un factor asociado con mayor productividad (medida a través del número de ventas) dado el p -valor significativo y el signo positivo del coeficiente. Por otra parte, en comparación con los empleados cuyo nivel educativo máximo al momento de la contración es la secundaria completa, aquellos con estudios universitarios incompletos están asociados con niveles más bajos de productividad (p-valor significativo y signo negativo del coeficiente).

De acuerdo a los datos se sugiere preferir candidatos con mayor experiencia y cuyo nivel de estudios sea diferente a universitarios incompletos.

[El modelo lineal requiere de validaciones adicionales sobre los supuestos que el científico de datos en su proyecto debe tener en cuenta]

Los estudiantes en su proyectos de la actividad 4 pueden trabajar en desarrollar más este ejemplo o las otras ideas de problemas especificos que les surgan.

In [ ]: