DATA ANALYST: CROSS SELLING RECOMMENDATION FINAL PROJECT

TEAM MEMBER'S DETAILS

Group Name: Individual

Name: Ian Kihara Wangui

Email: eandavid6@gmail.com

Company: DataGlacier

Specialization: Data Analyst

PROBLEM STATEMENT

How to increase cross selling of Banking Products

XYZ credit union in Latin America is performing very well in selling the Banking products (eg: Credit card, deposit account, retirement account, safe deposit box etc) but their existing customer is not not buying more than 1 product which means bank is not performing good in cross selling (Bank is not able to sell their other offerings to existing customer). XYZ Credit Union decided to approach ABC analytics to solve their problem.

Can you tell us how this can be solved?

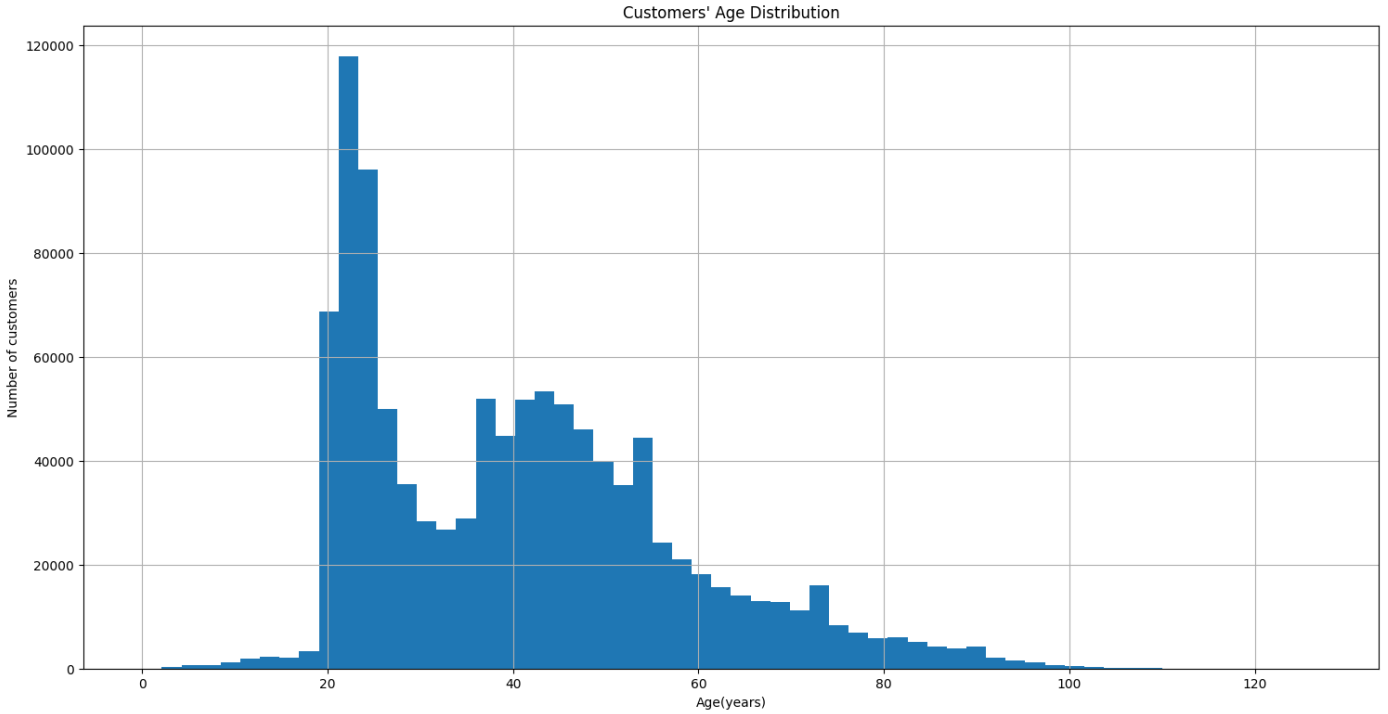
GITHUB REPO LINK

https://github.com/Iandavidk/DataGlacier/tree/main/Week-10-Cross_Selling_EDA

EXPLORATORY DATA ANALYSIS

1. Age distribution of customers

In [17]:
```
# Age distribuion of customers
plt.figure(figsize=(18,9))
df['customer_age'].hist(bins=60)
plt.title("Customers' Age Distribution")
plt.xlabel("Age(years)")
plt.ylabel("Number of customers") ;
```



Insight 1

It's interesting that the distribution is bimodal. There are a large number of university aged students, and then another peak around middle-age.

2. Customer attraction by channel

In [18]:
```
# Customer attraction by channel
df['channel_used_join'].value_counts(normalize=True).head(15)
```

Out[18]:
```
KHE    0.306298
KAT    0.255559
KFC    0.239331
KFA    0.033062
KHQ    0.022468
KHK    0.017727
KHD    0.008926
KAS    0.006966
KHM    0.006582
KAG    0.006272
RED    0.005378
KAA    0.005255
KHN    0.005210
KAY    0.005118
KAB    0.005019
Name: channel_used_join, dtype: float64
```

Insight 2

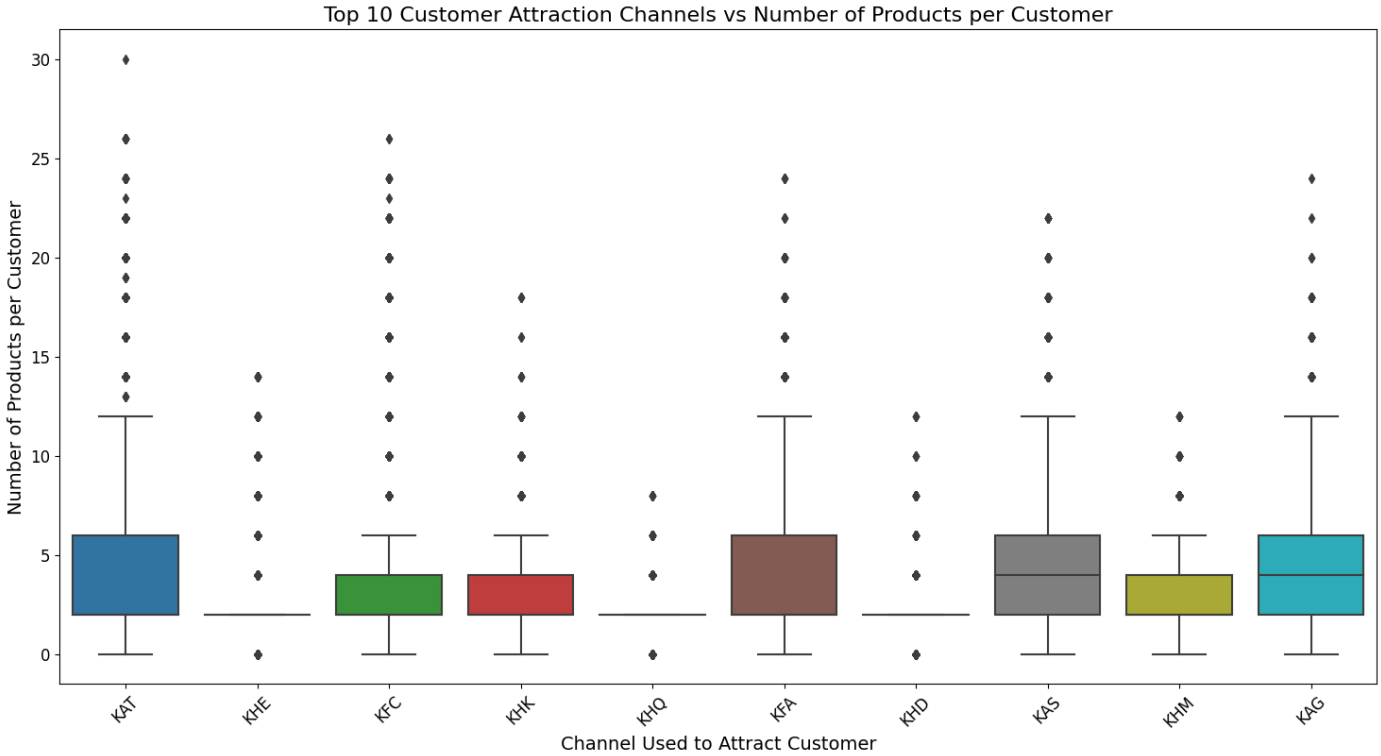
We can see that the most popular customer attraction channels are : 'KHE', 'KAT' and 'KFC'.

3. Relationship between the customer attraction channel and the number of products per customer

In [19]:
```
# Examine the relationship between the customer attraction channel and the number of pro
import seaborn as sns

top_10_channels = df['channel_used_join'].value_counts().head(10).index.tolist()

plt.figure(figsize=(18,9))
sns.boxplot(x='channel_used_join', y=df.iloc[:, 23:].sum(axis=1), data=df[df['channel_us
plt.title("Top 10 Customer Attraction Channels vs Number of Products per Customer", font
plt.xlabel("Channel Used to Attract Customer", fontsize=14)
plt.ylabel("Number of Products per Customer", fontsize=14)
plt.xticks(rotation=45, fontsize=12)
plt.yticks(fontsize=12) ;
```



Insight 3

The boxplot above shows the relationship between the top 10 customer attraction channels and the number of products per customer. It appears that customers who were attracted through channels such as 'KHE', 'KAT', and 'KFC' tend to have a higher number of products per customer compared to those who were attracted through other channels. However, it's important to note that there are outliers in each channel group, indicating that there are customers who have a high number of products regardless of the channel used to attract them. Overall, this visualization suggests that the customer attraction channel may have some influence on the number of products per customer, but it's not the only factor at play.

4. Total number of products per customer

In [20]:
```
# Total number of products per customer
df["total_products"].value_counts()
```

Out[20]:
```
1.0     556985
0.0     190776
2.0     159808
3.0      65472
4.0      38710
5.0      25443
6.0      18613
7.0      12570
8.0       6835
9.0       3041
10.0      1201
11.0       402
12.0       105
13.0        24
15.0         1
Name: total_products, dtype: int64
```

Insight 4

Most of the customers used one or two products and rarely use more than five products.

5. Total number of products by age

In [21]:
```
# Total number of products by age
tdf = df.groupby(['customer_age'])['total_products'].agg('sum')
tdf.sort_values(ascending=False).head(20)
```

Out[21]:
```
customer_age
23    57700.0
24    57026.0
43    54573.0
45    54491.0
44    54077.0
46    52090.0
42    51435.0
22    51247.0
47    49131.0
41    48133.0
48    47365.0
40    43395.0
49    42373.0
21    41873.0
50    39621.0
39    38215.0
25    37562.0
51    36600.0
52    34738.0
53    33468.0
Name: total_products, dtype: float64
```

In [22]:
```
# Number of products by age
plt.figure(figsize=(18,11))
tdf.plot(kind='bar', colormap='cool', legend=None)
plt.xticks(np.arange(0, 120, 10), [str(x) for x in np.arange(0, 120, 10)])
plt.title('Number of products by age')
plt.xlabel('Age(years)')
plt.ylabel('Number of products');
```

Insight 5

As we see bimodal distribution with most of the products used by middle aged customers between 35 and 55 years old, followed by young customers in thier twenties.

6. Number of products by customer regularity

In [23]:
```
# Number of products by customer regularity
tdf = df.groupby(['primary_customer_index'])['total_products'].agg('count')
tdf
```

Out[23]:
```
primary_customer_index
0.0           3
1.0     1079207
99.0        776
Name: total_products, dtype: int64
```

Insight 6

Almost all customers are primary customers throughout the month.

7. Number of products by "Customer relation type at the beginning of the month

In [24]:
```
# Number of products by "Customer relation type at the beginning of the month"
tdf = df.groupby(['customer_relation_beginning_month'])['total_products'].agg('count')
tdf
```

Out[24]:
```
customer_relation_beginning_month
0         3
A    501269
I    578713
P         1
Name: total_products, dtype: int64
```

Insight 7

Almost all customers are separated between active and inactive groups.

8. Total number of products by gross_income_household

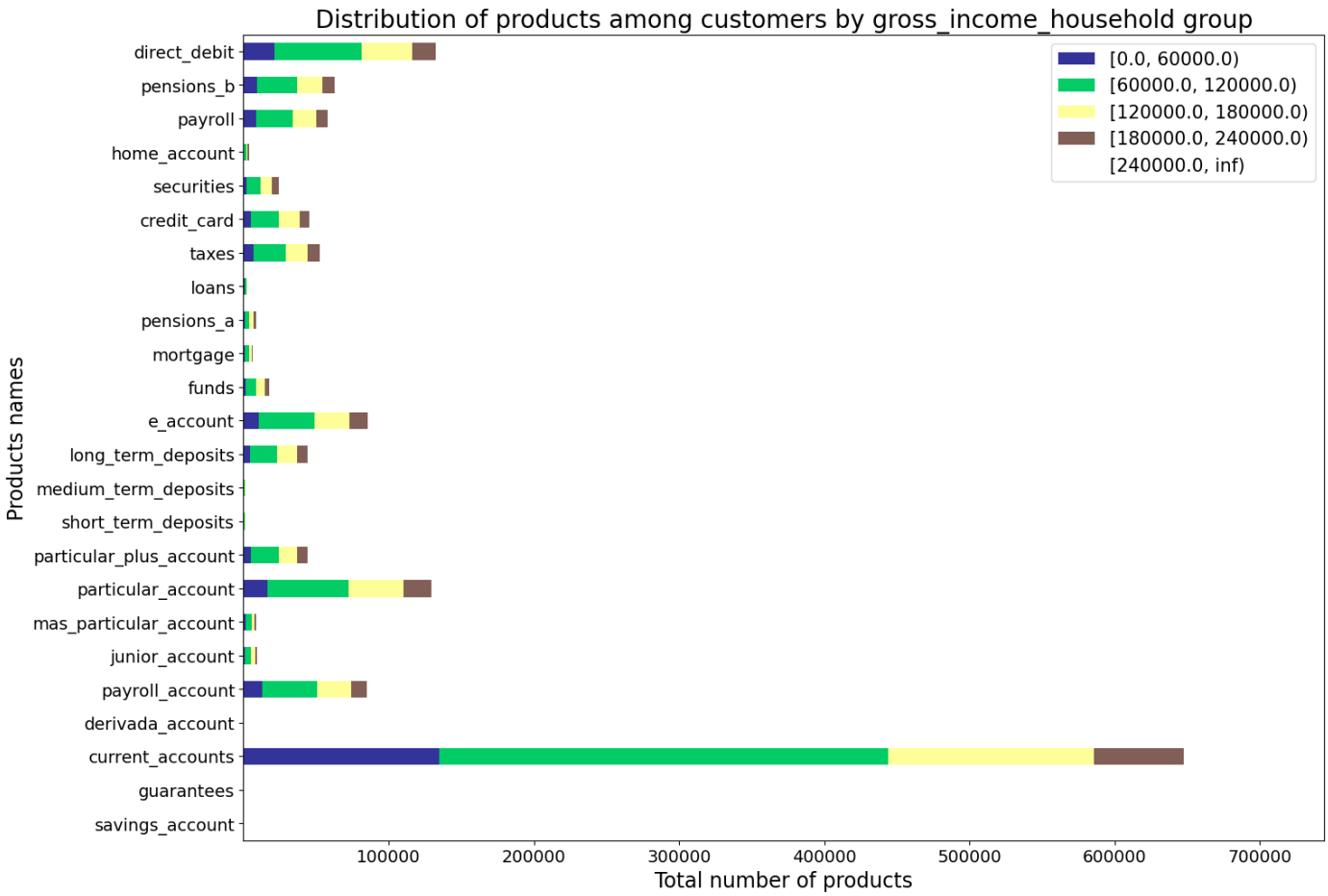
In [26]:
```
# Total number of products by gross_income_household

# Create income groups
tdf = (df.groupby(pd.cut(df['gross_income_household'], [0,60000,120000,180000,240000, pd
tdf = tdf.loc[:, "savings_account": "direct_debit"]
# Total products by income group
tdf
```

Out[26]:

gross_income_household	savings_account	guarantees	current_accounts	derivada_account	payroll_account	ju
[0.0, 60000.0)	13	2	134711	58	12846	
[60000.0, 120000.0)	23	6	309313	145	37705	
[120000.0, 180000.0)	27	5	141698	105	23373	
[180000.0, 240000.0)	23	11	61515	59	11052	
[240000.0, inf)	21	7	61712	51	10593	

In [27]:
```
# Plot of product share for each gross_income_household group
tdf =tdf.transpose()
tdf.plot(kind='barh', stacked=True, fontsize=14, figsize=[16,12], colormap='terrain')
plt.title('Distribution of products among customers by gross_income_household group', fo
plt.xlabel('Total number of products', fontsize=17, color='black')
plt.ylabel('Products names', fontsize=17, color='black')
plt.legend(prop={'size':15}, loc=1);
```



Insight 8

The visualization above shows the distribution of products among customers by gross_income_household group. The income groups are divided into 5 categories: 0-60,000, 60,000-120,000, 120,000-180,000, 180,000-240,000, and above 240,000.

From the visualization, we can infer that customers with higher gross income tend to have more products with the bank. The (60,000-120,000) group has the highest number of products, while the group with the lowest gross income (above 240,000) has the lowest number of products.

The most popular products among all income groups are current accounts, followed by particular accounts and direct debit accounts. The least popular products are securities and mortgages, which are only held by a small percentage of customers across all income groups.

9. Total number of products by segmentation

In [28]:
```
# Total number of products by segmentation
tdf = df.groupby(['segmentation'])['total_products'].agg('sum')
tdf
```

Out[28]:
```
segmentation
0                          0.0
01 - TOP                201691.0
02 - PARTICULARES      1061074.0
03 - UNIVERSITARIO      392043.0
Name: total_products, dtype: float64
```

Insight 9

PARTICULARES are the most important group

10. Total number of products by customer index

In [29]:
```
# Total number of products by customer index
tdf = df.groupby(['new_customer_index'])['total_products'].agg('count')
tdf
```

Out[29]:
```
new_customer_index
0.0    1054153
1.0      25833
Name: total_products, dtype: int64
```

Insight 10

Most customers are recurrent customers, older than six months

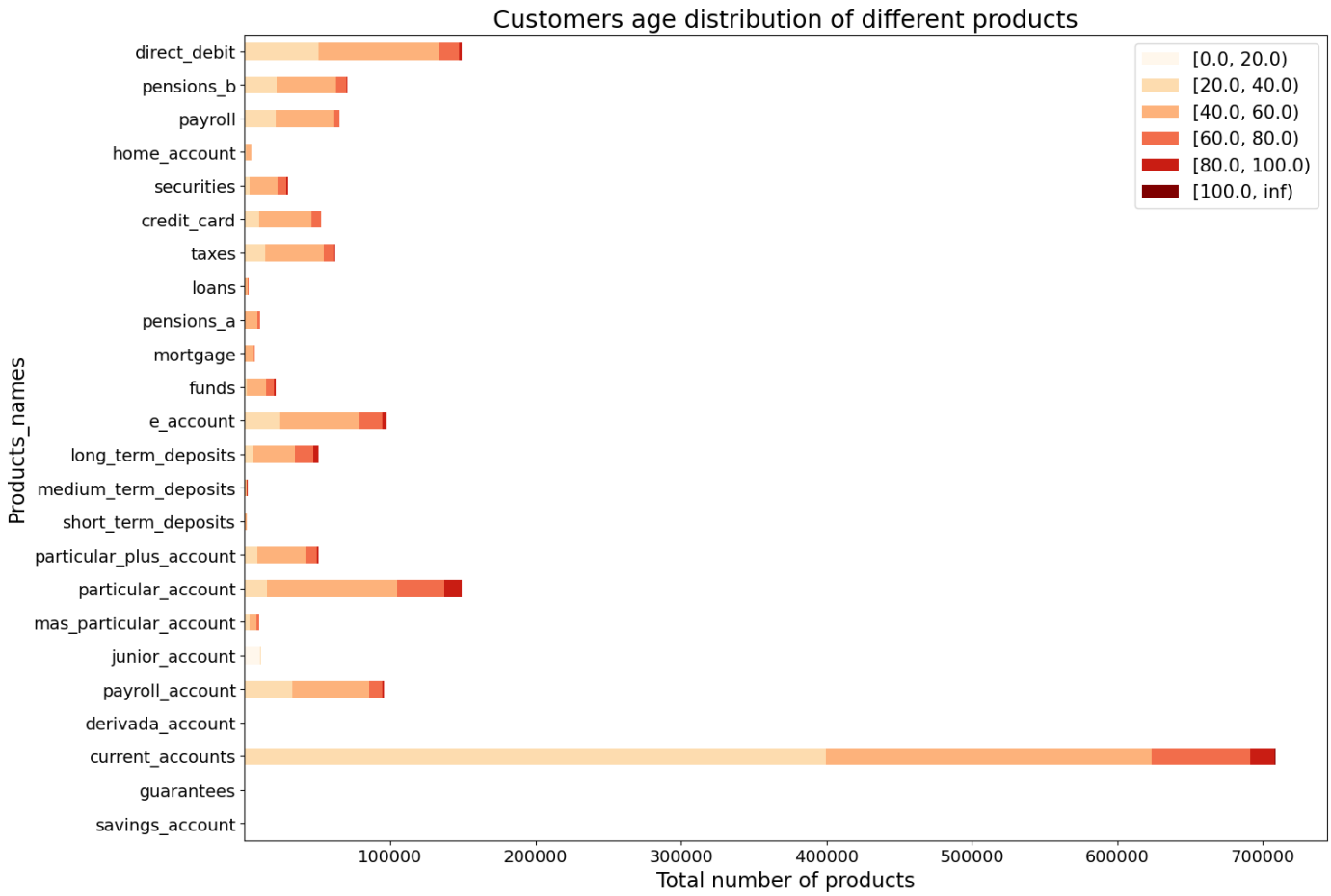
11. Total number of products by age group

In [30]:
```
# Total number of products by age group
tdf = (df.groupby(pd.cut(df['customer_age'], [0,20,40,60,80,100, pd.np.inf], right=False
tdf = tdf.loc[:, "savings_account": "direct_debit"]
tdf
```

Out[30]:

customer_age	savings_account	guarantees	current_accounts	derivada_account	payroll_account	junior_accour
[0.0, 20.0)	0	0	495	0	1	1080
[20.0, 40.0)	3	9	398815	48	32519	11
[40.0, 60.0)	94	22	224059	284	53319	
[60.0, 80.0)	10	0	67910	74	8489	
[80.0, 100.0)	0	0	17117	12	1233	
[100.0, inf)	0	0	553	0	8	

In [31]:
```
# Plot of customers' age distibution of each product
tdf =tdf.transpose()
tdf.plot(kind='barh', stacked=True, fontsize=14, figsize=[16,12], colormap='OrRd')
plt.title('Customers age distribution of different products', fontsize=20, color='black'
plt.xlabel('Total number of products', fontsize=17, color='black')
plt.ylabel('Products_names', fontsize=17, color='black')
plt.legend(prop={'size':15}, loc=1);
```



Insight 11

The visualization above shows the distribution of customers' age for each product. From the plot, we can infer that:

- The majority of customers who have a mortgage are between 40 and 60 years old.
- Customers who have a payroll account, pensions, and direct debit are mostly between 20 and 60 years old.
- Customers who have short-term deposits, medium-term deposits, and long-term deposits are mostly between 40 and 80 years old.
- Customers who have securities are mostly between 60 and 80 years old.
- Customers who have e-account, funds, and credit card are mostly between 20 and 40 years old.
- Customers who have home account, taxes, and loans are mostly between 20 and 60 years old.

12. Total number of products by province name amd province code

In [32]:
```
# Total number of products by province name amd province code
tdf = df.groupby(['province_name', 'province_code'])['total_products'].agg('count')
tdf = tdf.sort_values(ascending=False)
tdf.head(20)
```

Out[32]:
```
province_name  province_code
MADRID         28               390781
BARCELONA      8                106821
VALENCIA       46                54638
SEVILLA        41                51255
CORUÑA, A      15                31015
ZARAGOZA       50                29091
MALAGA         29                28771
MURCIA         30                28746
ALICANTE       3                 23475
CADIZ          11                23175
VALLADOLID     47                21187
ASTURIAS       33                20024
PONTEVEDRA     36                19638
PALMAS, LAS    35                17683
BADAJOZ        6                 15130
TOLEDO         45                14746
GRANADA        18                14091
SALAMANCA      37                12966
CORDOBA        14                11597
CANTABRIA      39                11191
Name: total_products, dtype: int64
```

Insight 12

The output above shows the total number of products by province name and province code. The table is sorted in descending order, and the top 20 provinces with the highest number of products are displayed. From the table, we can infer that:

- Madrid (28) has the highest number of products, followed by Barcelona (8) and Valencia (46).
- The top 20 provinces with the highest number of products are mostly located in the central and eastern regions of Spain.
- The number of products in each province varies widely, with Madrid having over 100,000 products and the 20th province having around 10,000 products.

13. Distribution of products by segment

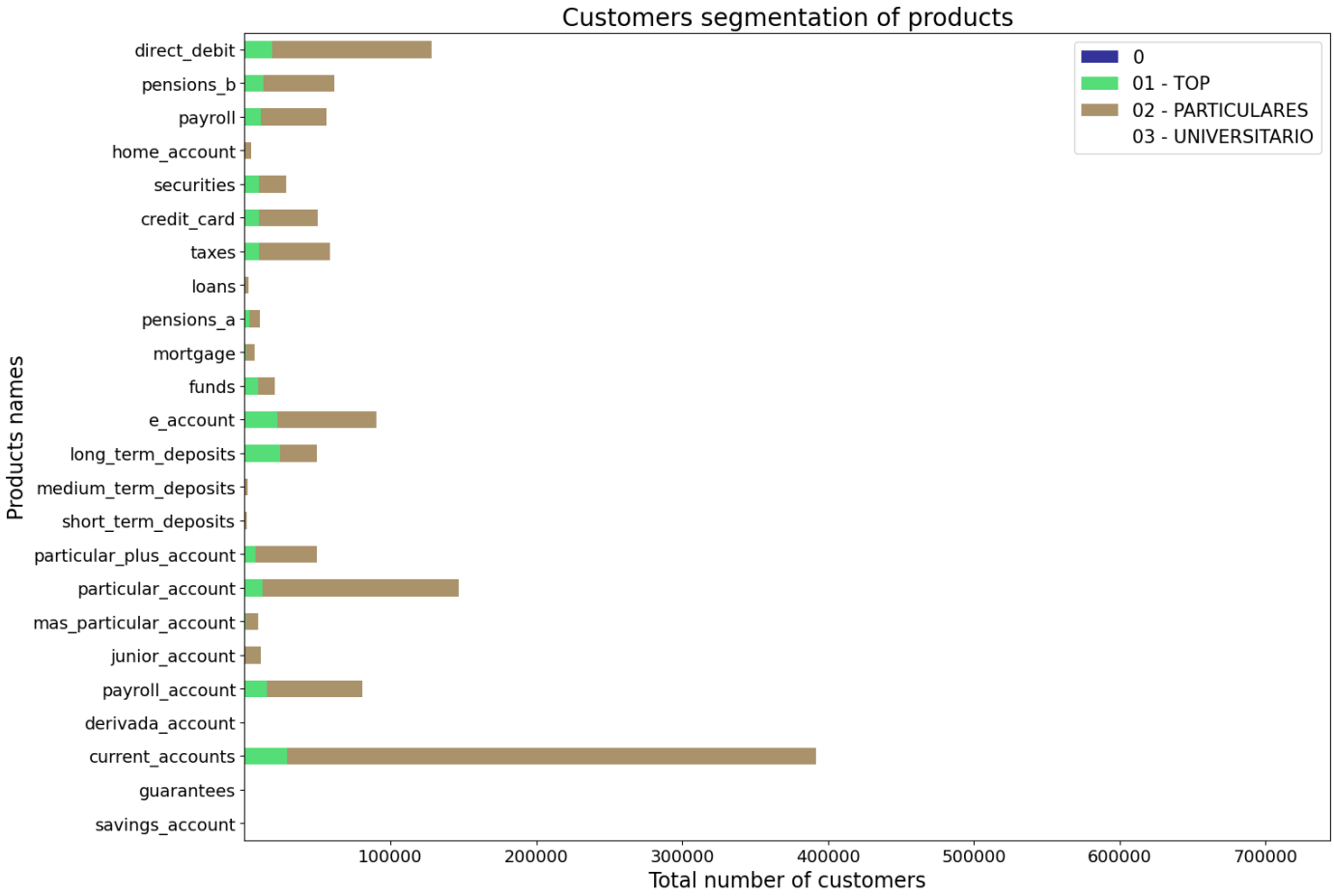
In [33]:
```
# Distribution of products by segment
tdf = df.loc[:, ['segmentation']].join(df.loc[:, 'savings_account':'direct_debit'])
tdf = tdf.groupby("segmentation").agg("sum")
tdf
```

Out[33]:

segmentation	savings_account	guarantees	current_accounts	derivada_account	payroll_account	junior_acco
0	0	0	0	0	0	
01 - TOP	8	11	29109	77	15447	
02 - PARTICULARES	99	20	362512	337	65449	10
03 - UNIVERSITARIO	0	0	317328	4	14673	

In [34]:
```
# Customers segment of each product
tdf = tdf.transpose()
tdf.plot(kind='barh', stacked=True, fontsize=14, figsize=[16,12], colormap='terrain')
plt.title('Customers segmentation of products', fontsize=20, color='black')
plt.xlabel('Total number of customers', fontsize=17, color='black')
```

```
plt.ylabel('Products names', fontsize=17, color='black')
plt.legend(prop={'size':15});
```



Insight 13

The visualization above shows the customers segmentation of products. From the stacked horizontal bar chart, we can infer that:

- The majority of customers for each product belong to the "02 - PARTICULARES" segment, followed by the "" segment.
- The "03 - UNIVERSITARIO" segment has the lowest number of customers for each product.
- The distribution of customers across segments varies for each product, with some products having a more even distribution across segments while others are dominated by a single segment.
- Overall, the "02 - PARTICULARES" segment has the highest number of customers across all products, followed by the "01 - TOP" segment.

14. Distribution of products by activity index

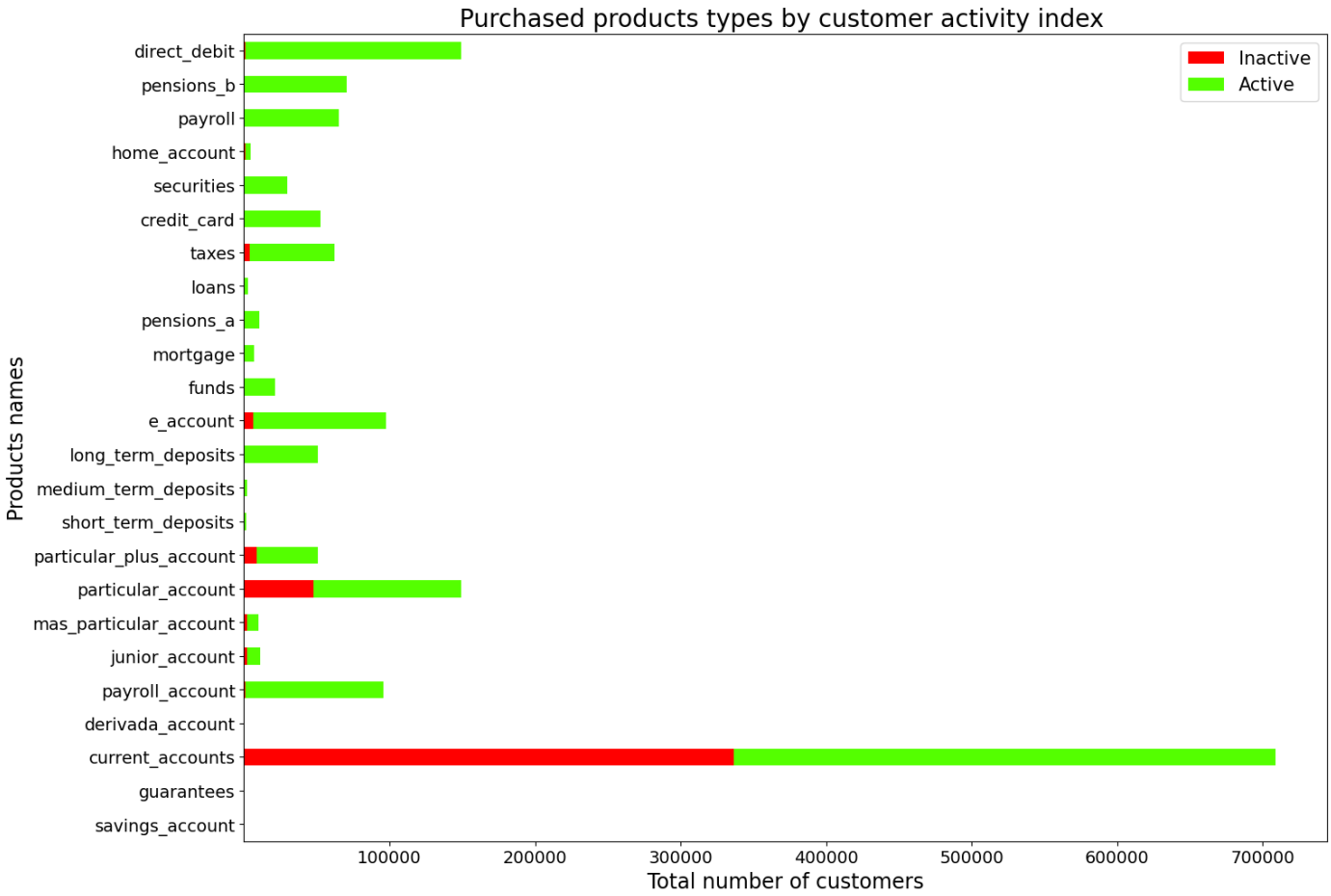
In [35]:
```
# Distribution of products by activity index
tdf = df.loc[:, ['activity_index']].join(df.loc[:, 'savings_account':'direct_debit'])
tdf = tdf.groupby("activity_index").agg("sum")
tdf
```

Out[35]:

activity_index	savings_account	guarantees	current_accounts	derivada_account	payroll_account	junior_accoun
0.0	26	0	336777	34	1135	197
1.0	81	31	372172	384	94434	895

In [36]:

```
# Purchased products types by customer activity index
tdf = tdf.transpose()
tdf.plot(kind='barh', stacked=True, fontsize=14, figsize=[16,12], colormap='prism')
plt.title('Purchased products types by customer activity index ', fontsize=20, color='bl
plt.xlabel('Total number of customers', fontsize=17, color='black')
plt.ylabel('Products names', fontsize=17, color='black')
plt.legend(["Inactive", "Active"], prop={'size':15});
```



Insight 14

From the visualization above, we can infer that the distribution of purchased products types varies by customer activity index. The "Active" customers have a higher number of purchased products across all product types compared to the "Inactive" customers.

The "Active" customers have a higher number of "long_term_deposits", "funds", "mortgage", "pensions_a", "loans", "credit_card", "securities", "home_account", "payroll", "pensions_b", and "direct_debit" products compared to the "Inactive" customers.

On the other hand, the "Inactive" customers have a higher number of "savings_account", "guarantees", "current_accounts", "derivada_account", "payroll_account", "junior_account", "mas_particular_account", "particular_account", "particular_plus_account", "short_term_deposits", "medium_term_deposits", and "e_account" products compared to the "Active" customers.

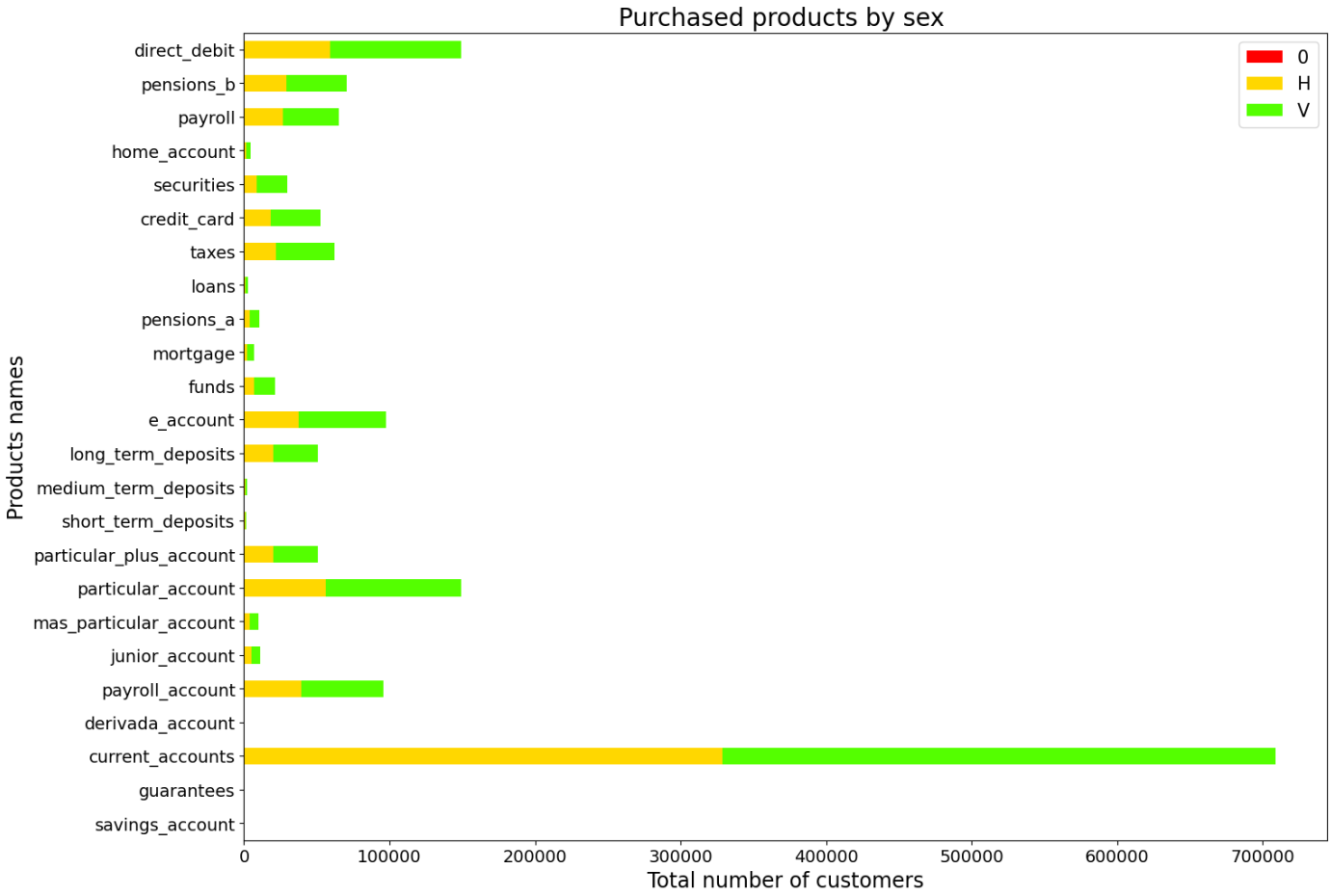
15. Distribution of products by sex

In [37]:
```
# Distribution of products by sex
tdf = df.loc[:, ['customer_sex']].join(df.loc[:, 'savings_account':'direct_debit'])
tdf = tdf.groupby("customer_sex").agg("sum")
tdf
```

Out[37]:

customer_sex	savings_account	guarantees	current_accounts	derivada_account	payroll_account	junior_accour
0	0	0	0	0	0	
H	26	12	328847	45	39474	537
V	81	19	380102	373	56095	554

In [38]:
```
# Total products purchased by sex
tdf = tdf.transpose()
tdf.plot(kind='barh', stacked=True, fontsize=14, figsize=[16,12], colormap='prism')
plt.title('Purchased products by sex ', fontsize=20, color='black')
plt.xlabel('Total number of customers', fontsize=17, color='black')
plt.ylabel('Products names', fontsize=17, color='black')
plt.legend(prop={'size':15});
```



Insight 15

From the visualization above, we can infer that the distribution of purchased products types varies by customer sex. The number of purchased products is higher for female customers compared to male customers across all product types.

Overall, the most purchased products by both male and female customers are "particular_account", "current_accounts", "e_account", and "direct_debit" products.

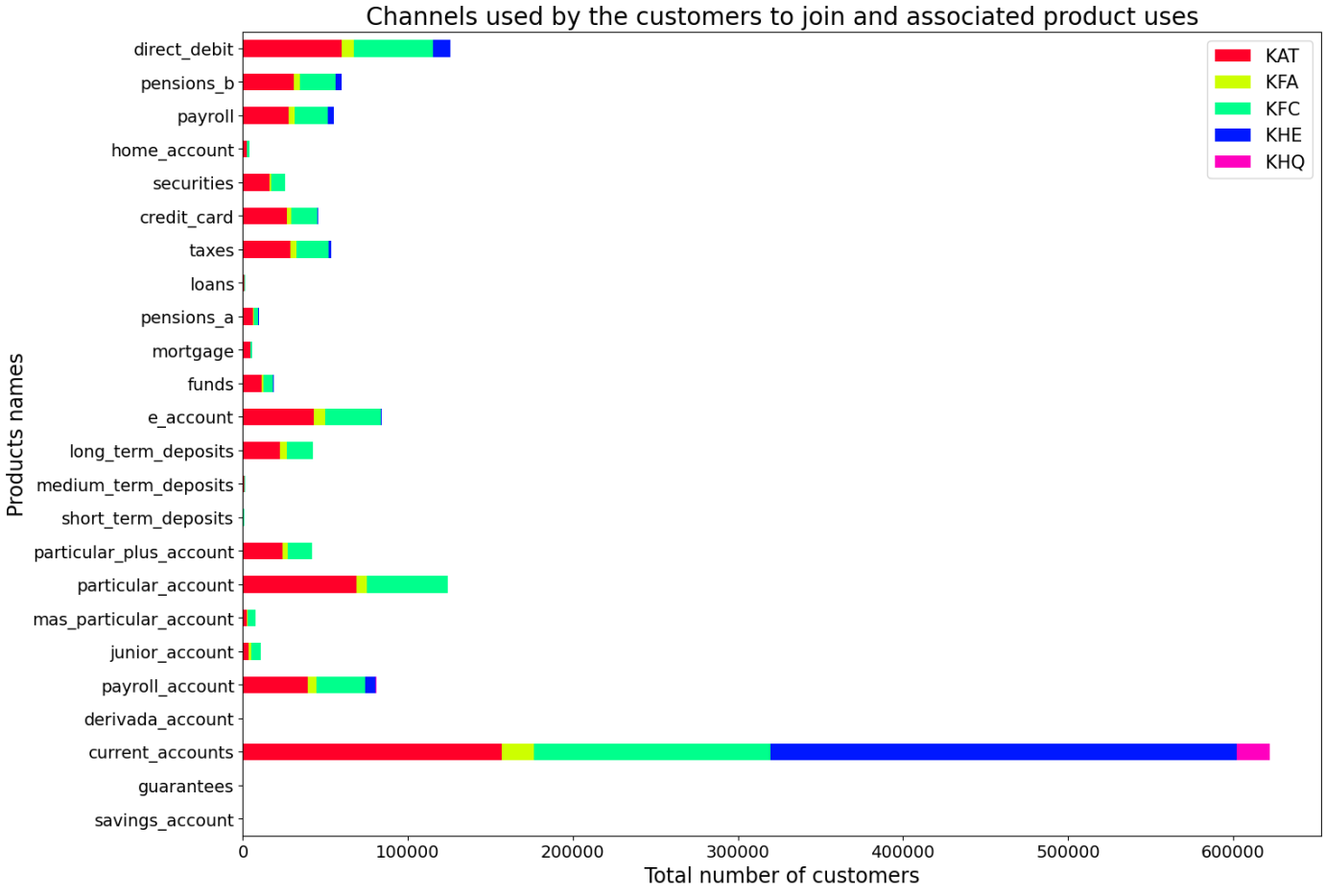
16. Distribution of products by channel used to join

In [39]:
```
# Distribution of products by channel used to join
tdf = df.loc[:, ['channel_used_join']].join(df.loc[:, 'savings_account':'direct_debit'])
subset = ["KHE", "KAT", "KFC", "KFA", "KHQ"]
tdf = tdf.loc[tdf['channel_used_join'].isin(subset)]
tdf = tdf.groupby("channel_used_join").agg("sum")
tdf
```

Out[39]:

channel_used_join	savings_account	guarantees	current_accounts	derivada_account	payroll_account	junior_ac
KAT	67	16	156654	204	39484	
KFA	2	0	19620	12	4932	
KFC	17	11	143249	143	29553	
KHE	0	0	282534	2	6713	
KHQ	0	0	20414	0	133	

In [40]:
```
# Channels used by the customer to join and the purchased products
tdf = tdf.transpose()
tdf.plot(kind='barh', stacked=True, fontsize=14, figsize=[16,12], colormap='gist_rainbow
plt.title('Channels used by the customers to join and associated product uses', fontsize
plt.xlabel('Total number of customers', fontsize=17, color='black')
plt.ylabel('Products names', fontsize=17, color='black')
plt.legend(prop={'size':15}) ;
```



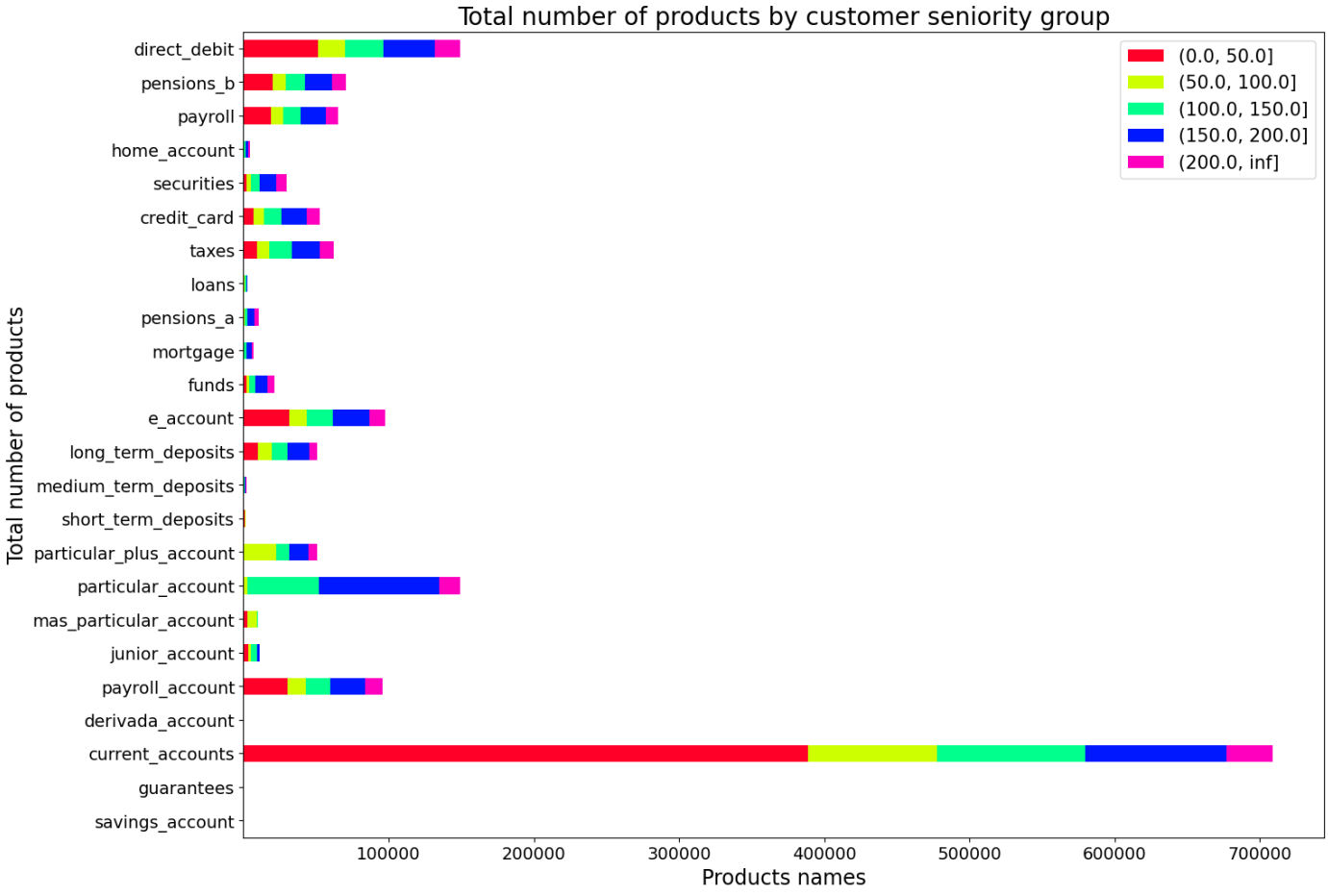
Insight 16

From the visualization above, we can infer that the channels used by customers to join the bank are associated with the types of products they purchase. Customers who joined through channels KHE, KAT, KFC, KFA, and KHQ tend to purchase more "particular_account", "current_accounts", "e_account", and "direct_debit" products compared to other products. This information can be useful for the bank to target specific products to customers based on the channels they used to join.

17. Distribution of products by customer seniority group

In [50]:
```
# Distribution of products by customer seniority group
tdf = df.loc[:, ['customer_seniority']].join(df.loc[:, 'savings_account':'direct_debit']
tdf['customer_seniority'] = pd.cut(tdf['customer_seniority'], bins=[0, 50, 100, 150, 200
tdf = tdf.groupby("customer_seniority").agg("sum")
tdf = tdf.transpose()

# Plotting stacked bar chart
tdf.plot(kind='barh', stacked=True, fontsize=14, figsize=[16,12], colormap='gist_rainbow
plt.title('Total number of products by customer seniority group', fontsize=20, color='bl
plt.xlabel('Products names', fontsize=17, color='black')
plt.ylabel('Total number of products', fontsize=17, color='black')
plt.legend(prop={'size':15}) ;
```



Insight 17

From the visualization above, we can infer that the total number of products purchased by customers is positively correlated with their seniority group. Customers who have been with the bank for a longer period tend to purchase more products compared to those who have been with the bank for a shorter period. This

information can be useful for the bank to target specific products to customers based on their seniority group.

FINAL RECOMMENDATIONS

Based on the exploratory data analysis section above, the following recommendations can be made to increase cross-selling of the banking products:

1. Target specific products to customers based on their seniority group.

Customers who have been with the bank for a longer period tend to purchase more products compared to those who have been with the bank for a shorter period. Therefore, the bank can target lower seniority groups with specific products that are likely to be of interest to them.

2. Offer incentives or discounts to customers who purchase multiple products.

This can encourage customers to purchase more products and increase cross-selling. Based on analysis above customers who have purchased one product are more likely to purchase another product. Therefore, offering incentives or discounts to customers who purchase multiple products can increase the likelihood of cross-selling.

3. Analyze the products that are frequently purchased together and create bundled packages to offer to customers.

This can increase the likelihood of customers purchasing multiple products. For example, based on the analysis above, customers who have a savings account are more likely to have a mortgage and a pension plan. Therefore, the bank can create bundled packages that include these products to encourage customers to purchase multiple products.

4. Provide personalized recommendations to customers based on their transaction history and purchase behavior.

This can increase the relevance of the recommendations and encourage customers to purchase more products. Based on the analysis above, customers who have a salary account are more likely to have a credit card and a direct debit. Therefore, the bank can provide personalized recommendations to customers who have a salary account based on their transaction history and purchase behavior to encourage them to purchase a credit card and a direct debit.

5. Conduct targeted marketing campaigns to promote specific products to customers who have not yet purchased them.

This can increase awareness of the products and encourage customers to try them out. The EDA analysis shows that there are some products that have low purchase rates, such as securities and funds. Therefore, the bank can conduct targeted marketing campaigns to promote these products to customers who have not yet purchased them.

Overall, these recommendations can help the bank increase cross-selling of its products and improve its revenue.