classification

2020 年 6 月 7 日

[2]:
```
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np
data = pd.read_csv('E:/pycharmProject/classification/284_618_compressed_vgsales/
 ↪vgsales.csv', encoding='utf-8')
path = 'E:/pycharmProject/classification/'
```

[44]:
```
def plot_bar(series_global_name, series_global_value,series_na_name,␣
 ↪series_na_value,series_eu_name, series_eu_value,series_jp_name,␣
 ↪series_jp_value, title):
    plt.clf()
    fig = plt.figure(figsize=(12,12))
    plt.subplot(221)
    tnt = 0
    number = []
    for values in series_global_value:
        number.append(float(values))
        tnt += 1
        if tnt == 25:
            break
    game_type = []
    tnt = 0
    for values in series_global_name:
        game_type.append(values)
        tnt += 1
        if tnt == 25:
            break
    x = np.arange(len(game_type))
```

```
    plt.bar(x, number)
    plt.xticks(x, game_type, rotation=90)
    plt.grid(axis='y', linestyle='--')
    plt.title(title+' Popularity by Global Sales')

    plt.subplot(222)
    tnt = 0
    number = []
    for values in series_na_value:
        number.append(float(values))
        tnt += 1
        if tnt == 25:
            break
    game_type = []
    tnt = 0
    for values in series_na_name:
        game_type.append(values)
        tnt += 1
        if tnt == 25:
            break
    x = np.arange(len(game_type))
    plt.bar(x, number)
    plt.xticks(x, game_type, rotation=90)
    plt.grid(axis='y', linestyle='--')
    plt.title(title + ' Popularity by NA Sales')

    plt.subplot(223)
    tnt = 0
    number = []
    for values in series_eu_value:
        number.append(float(values))
        tnt += 1
        if tnt == 25:
            break
    game_type = []
    tnt = 0
```

```
    for values in series_eu_name:
        game_type.append(values)
        tnt += 1
        if tnt == 25:
            break
    x = np.arange(len(game_type))
    plt.bar(x, number)
    plt.xticks(x, game_type, rotation=90)
    plt.grid(axis='y', linestyle='--')
    plt.title(title+' Popularity by European Sales')

    plt.subplot(224)
    tnt = 0
    number = []
    for values in series_jp_value:
        number.append(float(values))
        tnt += 1
        if tnt == 25:
            break
    game_type = []
    tnt = 0
    for values in series_jp_name:
        game_type.append(values)
        tnt += 1
        if tnt == 25:
            break
    x = np.arange(len(game_type))
    plt.bar(x, number)
    plt.xticks(x, game_type, rotation=90)
    plt.grid(axis='y', linestyle='--')
    plt.title(title +' Popularity by Japan Sales')
    fig.tight_layout()
    plt.savefig(path+'Most Popular '+title+'.jpg', bbox_inches='tight')
```

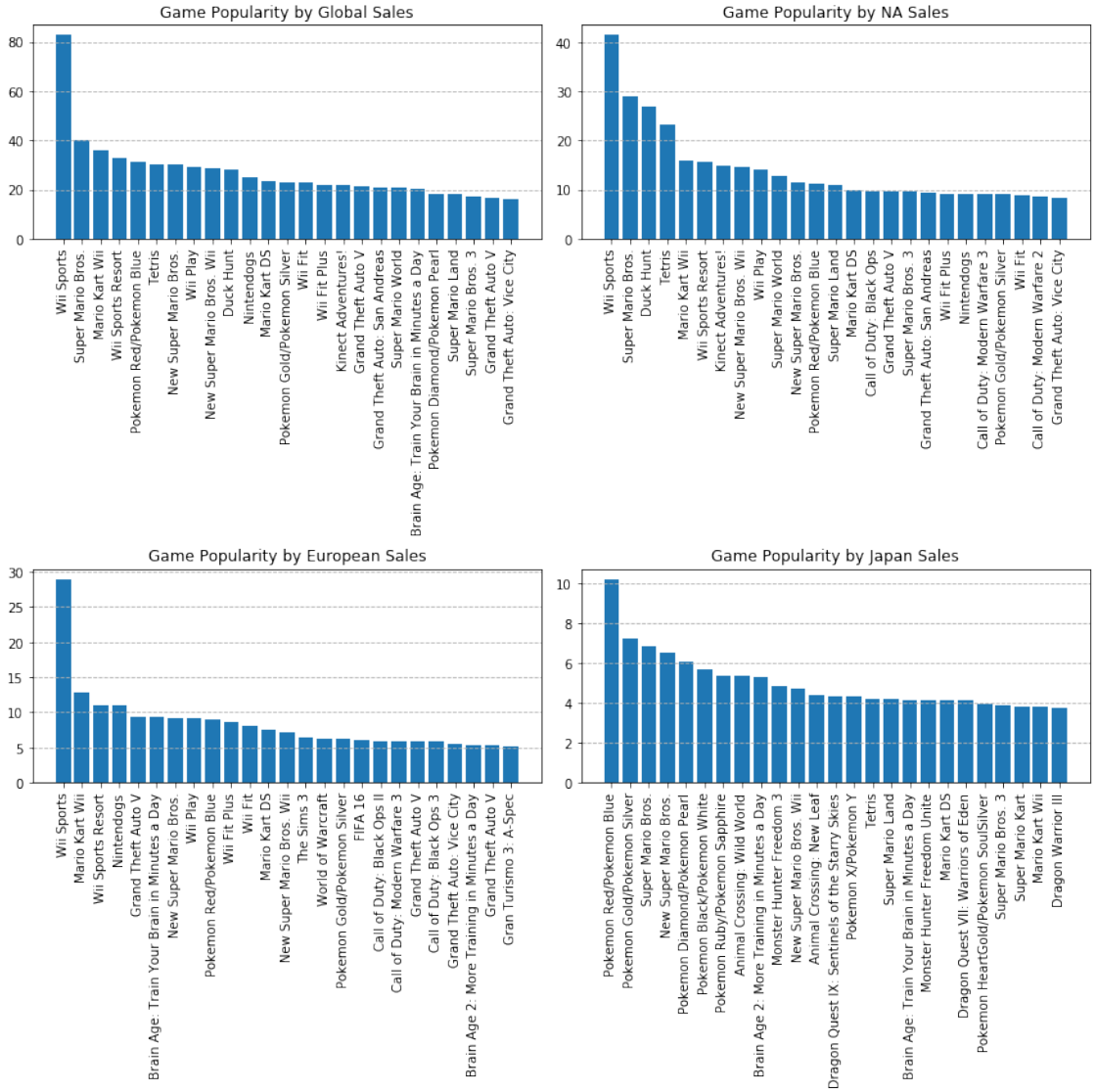
1 电子游戏市场分析

1.1 游戏受欢迎程度排名

popularity sort by global salse

[45]:
```
global_sales = data[['Name', 'Global_Sales']].sort_values(by='Global_Sales',␣
↪ascending=False)
na_sales = data[['Name', 'NA_Sales']].sort_values(by='NA_Sales',␣
↪ascending=False)
eu_sales = data[['Name', 'EU_Sales']].sort_values(by='EU_Sales',␣
↪ascending=False)
jp_sales = data[['Name', 'JP_Sales']].sort_values(by='JP_Sales',␣
↪ascending=False)
plot_bar(global_sales['Name'], global_sales['Global_Sales'], na_sales['Name'],␣
↪na_sales['NA_Sales'], eu_sales['Name'], eu_sales['EU_Sales'],␣
↪jp_sales['Name'], jp_sales['JP_Sales'],'Game')
```

```
<Figure size 432x288 with 0 Axes>
```



本文分别按照全球销量（global sales）、北美销量（na sales）、欧洲销量（eu sales）和日本销量（jp sales）对游戏的受欢迎程度进行了排名（仅显示前 25 名）。

可以看出，全球销量前三名的游戏是：

Wii Sports

Super Mario Bros.

Mario Kart Wii

北美销量前三名的游戏是：

Wii Sports

Super Mario Bros.

Duck Hunt

欧洲销量前三名的游戏是：

Wii Sports

Mario Kart Wii

Wii Sports Resort

日本销量前三名的游戏是：

Pokemon Red/Pokemon Blue

Pokemon Gold/Pokemon Silver

Super Mario Bros.

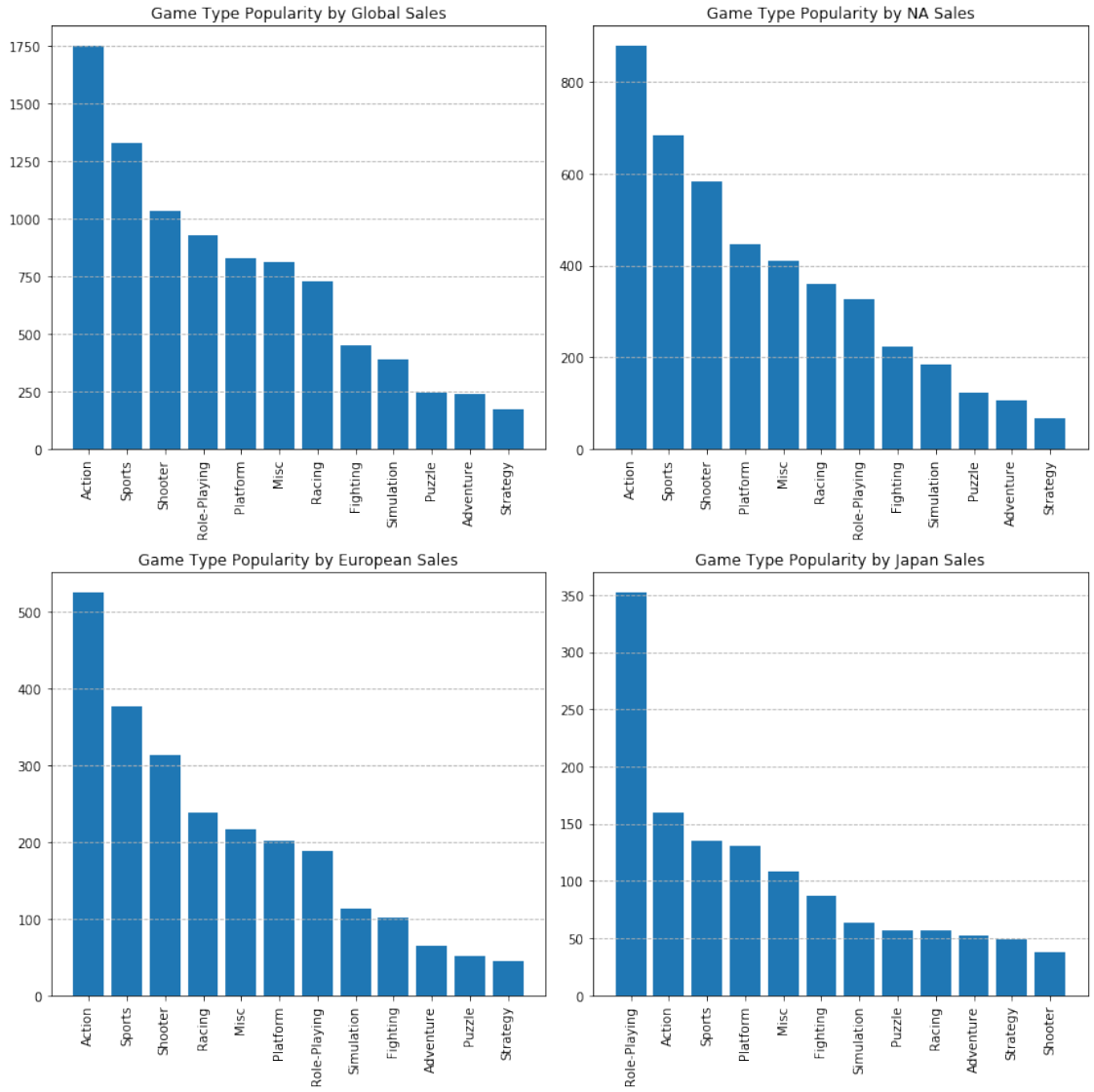
可以看出，Wii Sports 在欧美地区最受欢迎。Wii Sports 称为是任天堂史上最畅销游戏，他是一款体感游戏，需要身体的配合，比较符合喜爱户外运动的欧美人的喜好。但是 Wii Sports 在日本地区却没有那么受欢迎，可能和日本地区不喜欢激烈的体育运动的文化有关系。而 Pokemon 系列的游戏在日本尤为欢迎，这可能是因为宠物小精灵本身就是日本的国民 ip，在日本人的心中很受喜爱，同时这款游戏不需要剧烈的体育活动，因此在日本地区很受欢迎。

1.2 游戏类型受欢迎程度排名

[56]:
```
game_type_global = data[['Genre', 'Global_Sales']].
↪groupby(by=['Genre'])['Global_Sales'].sum().to_frame()
game_type_global = game_type_global.sort_values(by='Global_Sales',␣
↪ascending=False)

game_type_na = data[['Genre', 'NA_Sales']].groupby(by=['Genre'])['NA_Sales'].
↪sum().to_frame()
game_type_na = game_type_na.sort_values(by='NA_Sales', ascending=False)

game_type_eu = data[['Genre', 'EU_Sales']].groupby(by=['Genre'])['EU_Sales'].
↪sum().to_frame()
game_type_eu = game_type_eu.sort_values(by='EU_Sales', ascending=False)
```

```
game_type_jp = data[['Genre', 'JP_Sales']].groupby(by=['Genre'])['JP_Sales'].
 ↪sum().to_frame()
game_type_jp = game_type_jp.sort_values(by='JP_Sales', ascending=False)
plot_bar(game_type_global.index, game_type_global.values, game_type_na.index,␣
 ↪game_type_na.values, game_type_eu.index, game_type_eu.values, game_type_jp.
 ↪index, game_type_jp.values,'Game Type')
```

```
<Figure size 432x288 with 0 Axes>
```



通过最受欢迎的游戏类型分析可知，在欧美国家，最受欢迎的游戏类型前三位分别为：

Action、Sports、Shooter

均为比较激烈刺激类游戏。

在日本，最受欢迎的游戏类型前三类为：

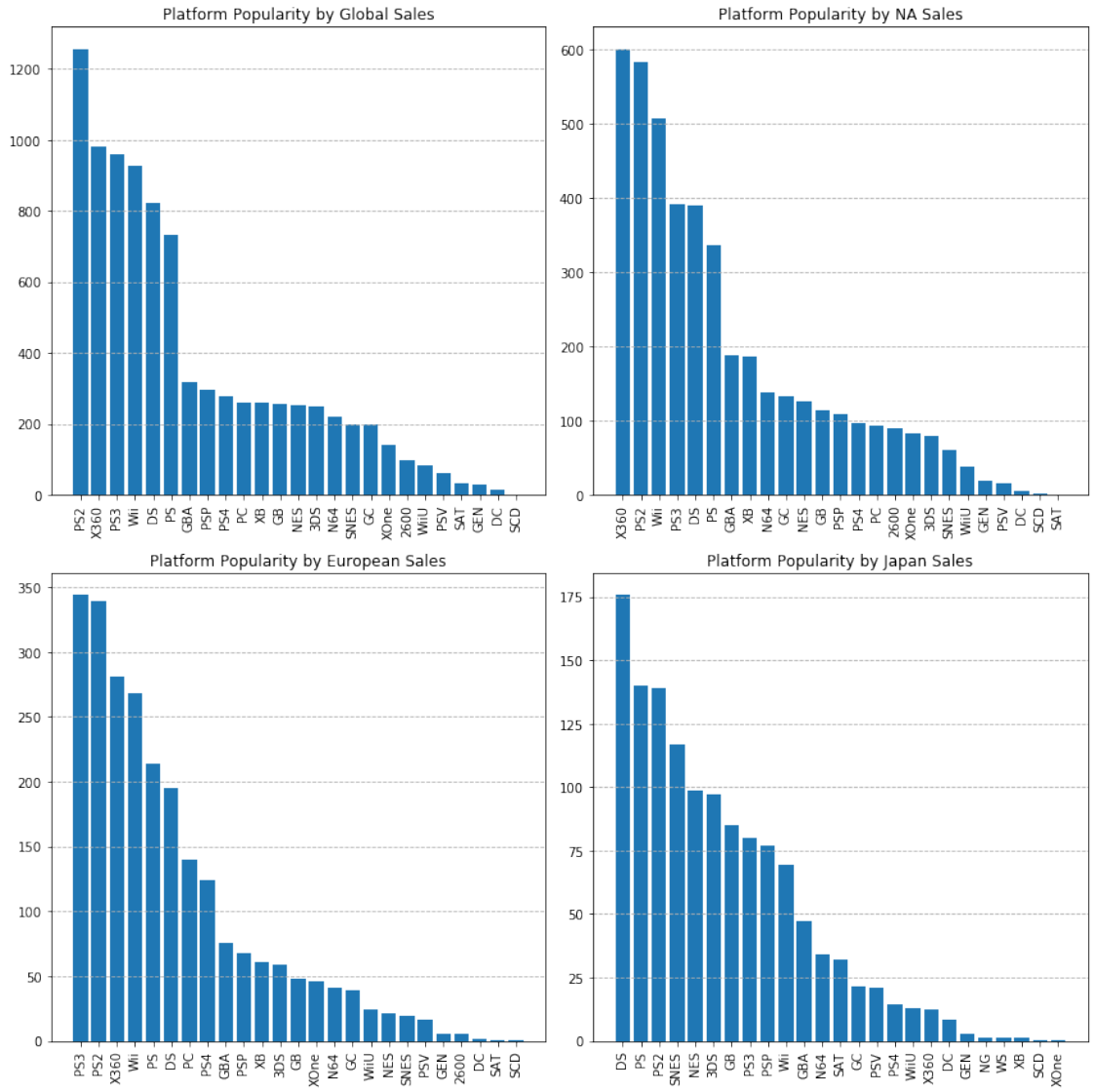
Role-Playing, Action, Sports

可见 Action 和 Sports 在世界范围内都很受欢迎。在日本 Role-Playing 类的游戏最受欢迎，这可能与日本发达的动漫产业密切相关。

1.3 游戏平台受欢迎程度排名

[57]:
```
platform_global = data[['Platform', 'Global_Sales']].
 ↪groupby(by=['Platform'])['Global_Sales'].sum().to_frame()
platform_global = platform_global.sort_values(by='Global_Sales',␣
 ↪ascending=False)

platform_na = data[['Platform', 'NA_Sales']].
 ↪groupby(by=['Platform'])['NA_Sales'].sum().to_frame()
platform_na = platform_na.sort_values(by='NA_Sales', ascending=False)

platform_eu = data[['Platform', 'EU_Sales']].
 ↪groupby(by=['Platform'])['EU_Sales'].sum().to_frame()
platform_eu = platform_eu.sort_values(by='EU_Sales', ascending=False)

platform_jp = data[['Platform', 'JP_Sales']].
 ↪groupby(by=['Platform'])['JP_Sales'].sum().to_frame()
platform_jp = platform_jp.sort_values(by='JP_Sales', ascending=False)
plot_bar(platform_global.index, platform_global.values, platform_na.index,␣
 ↪platform_na.values, platform_eu.index, platform_eu.values, platform_jp.
 ↪index, platform_jp.values,'Platform')
```

```
<Figure size 432x288 with 0 Axes>
```



可以看到在全球范围内，最受欢迎的游戏发布平台前三名分别是：

PS2

X360

PS3

在北美，最受欢迎的游戏发布平台前三名是：

X360

PS2

Wii

在欧洲，最受欢迎的游戏发布平台前三名是：

PS3

PS2

X360

在日本，最受欢迎的游戏发布平台前三名是：

DS

PS

PS2

经过查阅资料，PS2 游戏发布平台销量之所以能够成为第一，主要取决于一下几个历史条件：

1）游戏移植成本高，而日本厂商势力大，因此大部分游戏都是 PS2 独占

2）游戏人口经过 PS 时代的爆发性增长，达到高峰，尤其是欧洲市场刚被索尼统一，玩家的消费能力很高

3）游戏的收入和成本处于比较健康的平衡状态，游戏数量多

4）同期没有其他娱乐能和游戏竞争，连社交网络都没有普及

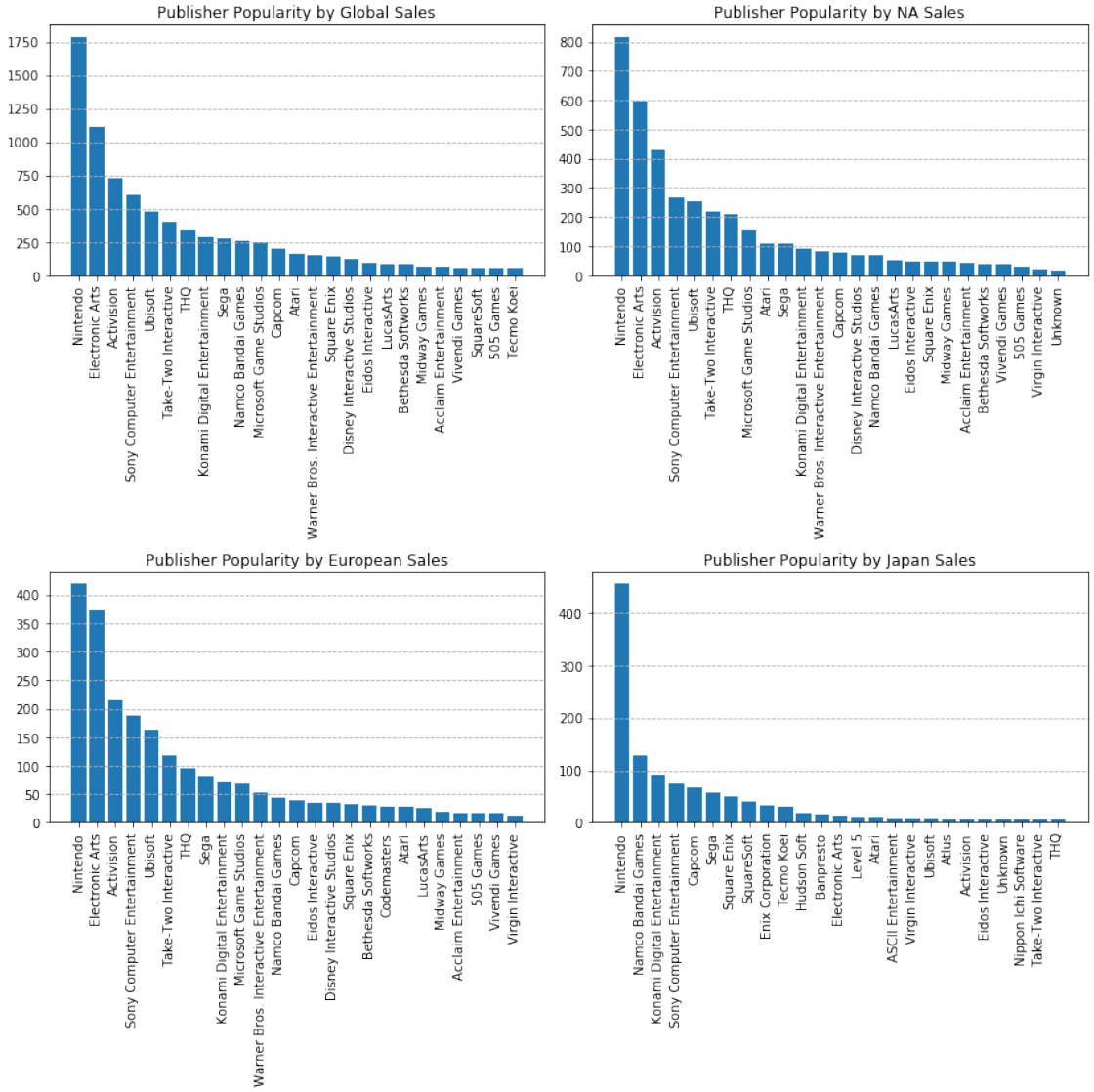
5）PS2 发布时，正好赶上 DVD 换代，而 PS2 可以一机多用，因此销量提升很大

1.4 游戏发布者受欢迎程度排名

[58]:
```
publisher_global = data[['Publisher', 'Global_Sales']].
 ↪groupby(by=['Publisher'])['Global_Sales'].sum().to_frame()
publisher_global = publisher_global.sort_values(by='Global_Sales',␣
 ↪ascending=False)

publisher_na = data[['Publisher', 'NA_Sales']].
 ↪groupby(by=['Publisher'])['NA_Sales'].sum().to_frame()
publisher_na = publisher_na.sort_values(by='NA_Sales', ascending=False)
```

```
publisher_eu = data[['Publisher', 'EU_Sales']].
 ↪groupby(by=['Publisher'])['EU_Sales'].sum().to_frame()
publisher_eu = publisher_eu.sort_values(by='EU_Sales', ascending=False)

publisher_jp = data[['Publisher', 'JP_Sales']].
 ↪groupby(by=['Publisher'])['JP_Sales'].sum().to_frame()
publisher_jp = publisher_jp.sort_values(by='JP_Sales', ascending=False)
plot_bar(publisher_global.index, publisher_global.values, publisher_na.index,␣
 ↪publisher_na.values, publisher_eu.index, publisher_eu.values, publisher_jp.
 ↪index, publisher_jp.values,'Publisher')
```

```
<Figure size 432x288 with 0 Axes>
```



可以看到，在游戏发布者上，任天堂以绝对优势拿下了第一名。在网络上有一句流行语 “任天堂就是世界的主宰”，足以看到这家专门做游戏的公司在游戏玩家心中的位置。任天堂的企业文化一直保障着其做出真正好玩的游戏，一直在游戏的玩法上去创新，每一作都能拿出特别吸引人的玩法，游戏本身有活力，自然经久不衰。

而第二名的位置在不同地区排名不同。在欧美国家，第二名的游戏发布者为 Electronic Arts。Electronic Arts 为美国知名游戏厂商，可能对于欧美市场更为熟悉，所以在欧美地区受欢迎。而在日本，第二名的位置属于 Namco Bandai Games。Namco 公司发布过多款与 Pokemon 有关的游戏，因此在日本更受欢迎。

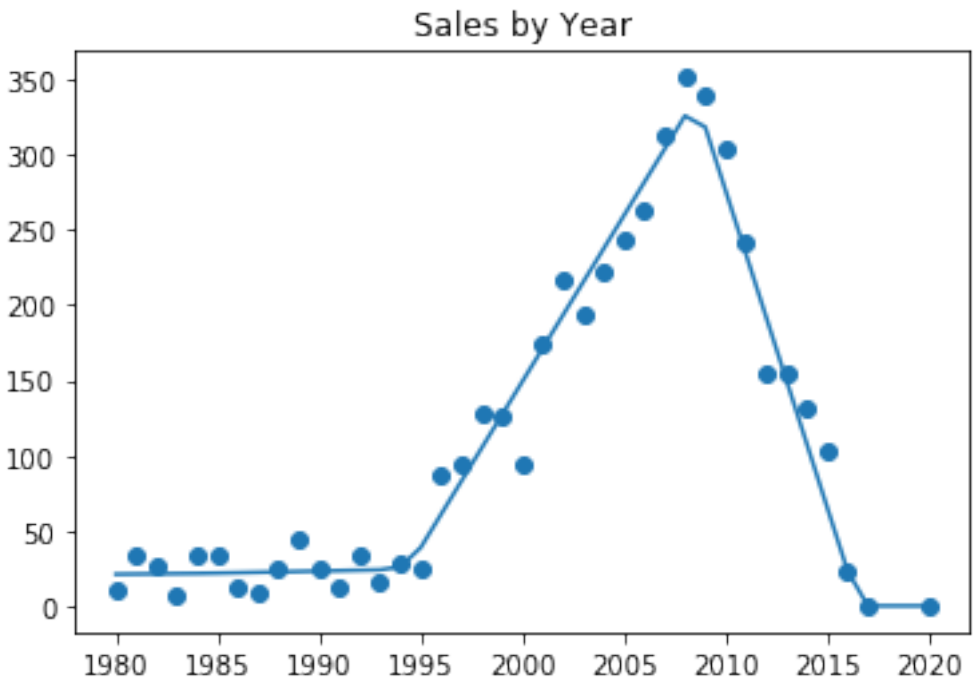
2 电子游戏销售额预测

[13]:
```
def plt_scatter(x_value, y_value, line_y):
    plt.clf()
    plt.scatter(x_value, y_value)
    plt.plot(x_value, line_y)
    plt.title('Sales by Year')
    plt.savefig(path+'sales_scatter.jpg')
```

[60]:
```
import torch.nn as nn
import torch
import torch.optim as optim

sales_global = data[['Year', 'NA_Sales']].groupby(by=['Year'])['NA_Sales'].
 ↪sum().to_frame()
sales_global = sales_global.sort_values(by='Year', ascending=False)
mx_year = np.max(sales_global.index)
mn_year = np.min(sales_global.index)
mx_sale = np.max(sales_global.values)
mn_sale = np.min(sales_global.values)
years = torch.tensor((sales_global.index-mn_year)/(mx_year-mn_year),␣
 ↪dtype=torch.float)
sales = torch.tensor((sales_global.values-mn_sale)/(mx_sale-mn_sale),␣
 ↪dtype=torch.float)
loss_weight = np.ones(len(sales))
mx_index = len(sales)
# print(sales[index])

# plt_scatter(sales_global.index, sales_global.values)




class Model(nn.Module):
    def __init__(self, trainable=True):
        super(Model, self).__init__()
        init_ = lambda m: self.init(m,
```

```
                        nn.init.orthogonal_,
                        lambda x: nn.init.constant_(x, 0))
        self.base = nn.Sequential(
            init_(nn.Linear(1, 64)),
            nn.ReLU(),
            init_(nn.Linear(64, 64)),
            nn.ReLU(),
            init_(nn.Linear(64, 1))
        )
        if trainable:
            self.train()
        else:
            self.eval()

    def init(self, module, weight_init, bias_init, gain=1):
        weight_init(module.weight.data, gain=gain)
        bias_init(module.bias.data)
        return module

    def get_sales(self, year):
        sale = self.base(year.unsqueeze(-1)).clamp(min=0)
        return sale.detach().squeeze()

    def get_loss(self, year_batch, sale_batch):
        pred_sale = self.base(year_batch)
        pred_sale = torch.clamp(pred_sale, min=0)
        forward_loss = (pred_sale - sale_batch).pow(2).sum() /2
        return forward_loss

model = Model()
training_time = 1500
sample_time = 30
tt = 0
loss_list = []
optimizer = optim.Adam(list(model.parameters()), lr=0.001)
while tt < training_time:
```

```
    sample_index = np.random.randint(0, mx_index, sample_time)

    _year_batch = years[sample_index].unsqueeze(-1)
    _sale_batch = sales[sample_index]
#     print(_year_batch)
    loss = model.get_loss(_year_batch, _sale_batch)
    model.zero_grad()
    loss.backward()
    loss_list.append(loss.clone().detach())
    optimizer.step()
    tt += 1
sample_index = np.arange(mx_index)
y_values = model.get_sales(years[sample_index].unsqueeze(-1))
y_values = y_values * (mx_sale - mn_sale) + mn_sale
# print(years[sample_index])
plt_scatter(sales_global.index, sales_global.values, y_values)
```



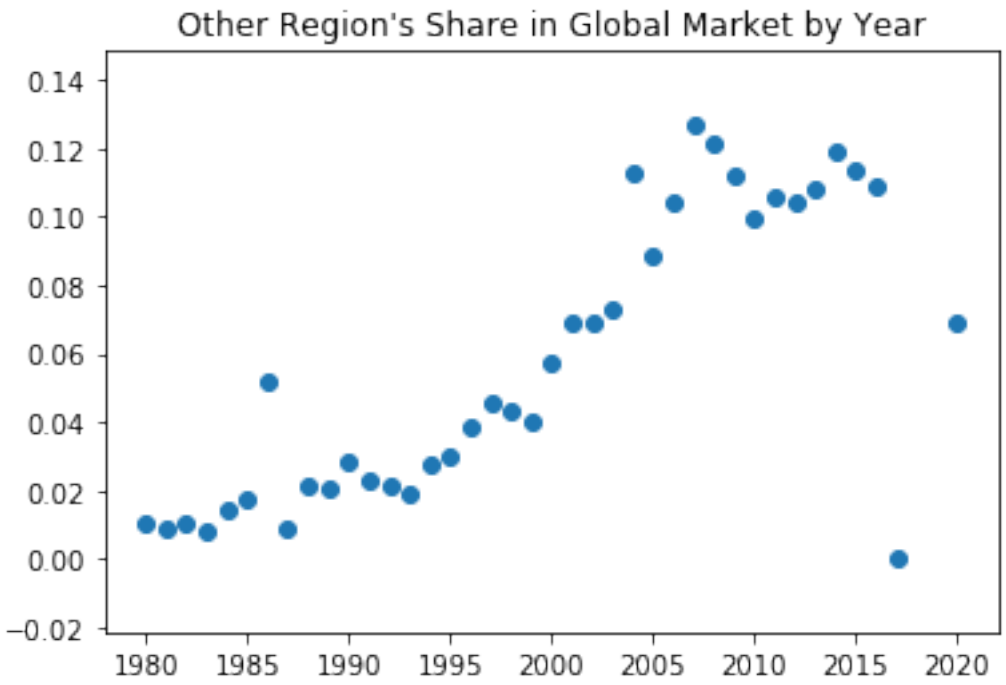
以北美游戏市场销量为例，统计从 1980 到 2020 年间，销量随着时间的变化情况。本文使用了深

度神经网络对数据进行了拟合和预测。

可以看出，北美市场的电子游戏销量在 1995 年前，基本以 4 年一个周期进行涨跌。销量在 1995 年迎来了巨大的提升，电子游戏市场吸引了众多的付费玩家，使得销量从 95 年开始到 08 年几乎一直在急速增加。从 2008 年开始，电子游戏市场的红利消失，销量在逐步降低，并且在 2017 年左右迎来了饱和。可以看出，如果保持当前态势，那么游戏市场不会有大的变动，游戏销售额基本稳定在一个较低的水平。

3 可视化应用

[66]:
```
def plt_scatter1(x_value, y_value):
    plt.clf()
    plt.scatter(x_value, y_value)
    plt.title('Other Region\'s Share in Global Market by Year')
    plt.savefig(path+'sales_ratio.jpg')

sales_global = data[['Year', 'Global_Sales']].
 ↪groupby(by=['Year'])['Global_Sales'].sum().to_frame()
sales_global = sales_global.sort_values(by='Year', ascending=False)

sales_other = data[['Year', 'Other_Sales']].groupby(by=['Year'])['Other_Sales'].
 ↪sum().to_frame()
sales_other = sales_other.sort_values(by='Year', ascending=False)

sales_ratio = [_other/_global for _other, _global in zip(sales_other.values,␣
 ↪sales_global.values)]
plt_scatter1(sales_global.index, sales_ratio)
```



尽管从市场分析中可以看出，今天年来全球的市场正在走低的阶段。但是通过分析市场占比，可以看出游戏市场将来可以挖掘的方向。从 1980 年以来，一直以欧美市场占据着游戏市场的大头比重。这是因为欧美市场由发达国家组成，国民收入较高，民众的需求从物质生活的需求转移到精神生活的需求，加上欧美的游戏技术较为发达，可以制造迎合欧美群众需要的游戏。而从其他地区的市场份额占比可以看出，近年来随着新型游戏市场的发展，其他地区在游戏市场中的占比逐年提高。这是因为随着经济全球化的发展，一些国家摆脱了贫困的生活，也开始了对于物质生活的追求，并且其他国家的人数占比要远远高于欧美发达国家。因此，如果游戏公司想要开发新的市场，应该着眼于亚非拉国家，特别是中国市场付费玩家的挖掘，开发适应于本土的游戏，从而实现在全球游戏市场走低的情况下，进行公司的盈利。

[ ]: