Preprocessing

In [ ]:
```
import numpy as np
import pandas as pd
```

In [ ]:
```
df = pd.read_csv("../data/final_0_80509.csv")
df
```

```
C:\Users\Lenovo\AppData\Local\Temp\ipykernel_5728\3731924929.py:1: DtypeWarning: Columns (10,35) have mixed types. Specify dtype option on import or set low_memory=False.
  df = pd.read_csv("../data/final_0_80509.csv")
```

Out[ ]:

	Index	AppID	Title	Initial_Price	Final_Price	Discount_Percent	Developers
0	0	20200	Galactic Bowling	NaN	NaN	NaN	['Perpetual FX Creative']
1	1	655370	Train Bandit	52.0	52.0	0.0	['Rusty Moyher']
2	2	1732930	Jolt Project	199.0	199.0	0.0	['Campião Games']
3	3	1355720	Henosis™	NaN	NaN	NaN	['Odd Critter Games']
4	4	1139950	Two Weeks in Painland	0.0	0.0	0.0	['Unusual Games']
...	...	...	...	...	...	...	...
80505	80505	574674	VRC PRO Deluxe Off-road tracks 4	570.0	285.0	50.0	['Virtual Racing Industries Ltd.']
80506	80506	947930	Car Mechanic Simulator 2018 - Porsche DLC	300.0	36.0	88.0	['Red Dot Games']
80507	80507	1900780	Erannorth Chronicles - Ancient Ruins	450.0	360.0	20.0	['Spyridon Thalassinos']
80508	80508	2470521	Crossout — Electric beetle (Lite edition)	1199.0	1199.0	0.0	['Targem Games']
80509	80509	2491770	We Need To Cook - Drug Empire Simulator	480.0	480.0	0.0	['Anark Studios', 'Crowbar Games']

80510 rows × 37 columns

In [ ]:
```
df.columns
```

Out[ ]:
```
Index(['Index', 'AppID', 'Title', 'Initial_Price', 'Final_Price',
       'Discount_Percent', 'Developers', 'Publishers', 'Genres', 'Categories',
       'Required_Age', 'Achievements', 'Release_Date', 'Metacritic_score',
       'DLC_Flag', 'Win_Flag', 'Mac_Flag', 'Linux_Flag', 'OS', 'Processor',
       'Memory', 'Graphics', 'DirectX', 'Storage', 'Current_Players',
       'Interface_Languages', 'Audio_Languages', 'Subtitle_Languages',
       'Positive_Reviews', 'Negative_Reviews', 'Total_Reviews',
       'Overall_Review_Summary', 'Recent_Reviews', 'Recent_Review_Summary',
       'Mature_Content_Desc', 'Awards', 'Curators'],
      dtype='object')
```

In [ ]:
```
```

In [ ]:
```
good_df = df[["Initial_Price", "Final_Price" , "Required_Age" , 'Win_Flag', 'Mac_Fl
good_df
```

Out[ ]:

	Initial_Price	Final_Price	Required_Age	Win_Flag	Mac_Flag	Linux_Flag	Memory	S
0	NaN	NaN	0	True	False	False	512 MB	
1	52.0	52.0	0	True	True	False	1 GB	
2	199.0	199.0	0	True	False	False	250 MB	
3	NaN	NaN	0	True	True	True	2 GB	
4	0.0	0.0	0	True	True	False	2 GB	
...	...	...	...	...	...	...	...	
80505	570.0	285.0	0	True	False	False	1 GB	
80506	300.0	36.0	0	True	True	False	4 GB	
80507	450.0	360.0	0	True	False	False	6 GB	
80508	1199.0	1199.0	0	True	False	False	4 GB	
80509	480.0	480.0	0	True	False	False	5 GB	

80510 rows × 11 columns

1. initial / final
2. Required_Age
3. 'Win_Flag', 'Mac_Flag', 'Linux_Flag'
4. mem and storage
5. +ve, -ve,

target overall_review_sumary

In [ ]:
```
classified = good_df[~good_df["Overall_Review_Summary"].isna()]
classified
```

Out[ ]:

	Initial_Price	Final_Price	Required_Age	Win_Flag	Mac_Flag	Linux_Flag	Memory	S
0	NaN	NaN	0	True	False	False	512 MB	
1	52.0	52.0	0	True	True	False	1 GB	
3	NaN	NaN	0	True	True	True	2 GB	
4	0.0	0.0	0	True	True	False	2 GB	
5	0.0	0.0	0	True	False	False	2 GB	
...	...	...	...	...	...	...	...	
80505	570.0	285.0	0	True	False	False	1 GB	
80506	300.0	36.0	0	True	True	False	4 GB	
80507	450.0	360.0	0	True	False	False	6 GB	
80508	1199.0	1199.0	0	True	False	False	4 GB	
80509	480.0	480.0	0	True	False	False	5 GB	

70727 rows × 11 columns

In [ ]:
```
discard = [np.nan, '5 user reviews',
       '1 user reviews',  '6 user reviews',
       '3 user reviews', '8 user reviews', 'No user reviews',
       '4 user reviews',
       '2 user reviews', '9 user reviews', '7 user reviews',
       ]
good_classified = classified[~classified["Overall_Review_Summary"].isin(discard)]
good_classified
```

Out[ ]:

	Initial_Price	Final_Price	Required_Age	Win_Flag	Mac_Flag	Linux_Flag	Memory	S
0	NaN	NaN	0	True	False	False	512 MB	
1	52.0	52.0	0	True	True	False	1 GB	
4	0.0	0.0	0	True	True	False	2 GB	
5	0.0	0.0	0	True	False	False	2 GB	
6	530.0	530.0	0	True	False	False	2 GB	
...	...	...	...	...	...	...	...	
80504	299.0	299.0	0	True	False	False	8 GB	
80506	300.0	36.0	0	True	True	False	4 GB	
80507	450.0	360.0	0	True	False	False	6 GB	
80508	1199.0	1199.0	0	True	False	False	4 GB	
80509	480.0	480.0	0	True	False	False	5 GB	

45365 rows × 11 columns

In [ ]:
```
bad_classified = classified[(classified["Overall_Review_Summary"].isin(discard) & ~
bad_classified
```

Out[ ]:

	Initial_Price	Final_Price	Required_Age	Win_Flag	Mac_Flag	Linux_Flag	Memory	S
3	NaN	NaN	0	True	True	True	2 GB	
7	349.0	349.0	0	True	False	False	1 GB	
12	85.0	85.0	0	True	False	False	NaN	
19	349.0	349.0	0	True	False	False	2 GB	
23	125.0	125.0	0	True	False	False	2 GB	
...	...	...	...	...	...	...	...	
80491	349.0	349.0	0	True	True	False	1 GB	
80498	1300.0	1300.0	0	True	False	False	8 GB	
80499	164.0	164.0	0	True	False	False	4 GB	
80502	610.0	610.0	0	True	False	False	4 GB	
80505	570.0	285.0	0	True	False	False	1 GB	

25362 rows × 11 columns

In [ ]:
```python
# target assign
def assign_target(row):
    if row["Overall_Review_Summary"] is  np.nan:
        return np.nan

    summary = row["Overall_Review_Summary"].lower()
    if "positive" in summary:
        return 1
    elif "negative" in summary:
        return -1
    elif "mixed" in summary:
        return 0

    p = row["Positive_Reviews"]
    n = row["Negative_Reviews"]

    if p > n:
        return 1
    elif p < n:
        return -1
    return 0

good_classified["target"] = good_classified.apply(assign_target,axis=1)
good_classified
```

```
C:\Users\Lenovo\AppData\Local\Temp\ipykernel_5728\380241893.py:23: SettingWithCopyWa
rning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/u
ser_guide/indexing.html#returning-a-view-versus-a-copy
  good_classified["target"] = good_classified.apply(assign_target,axis=1)
```

Out[ ]:

	Initial_Price	Final_Price	Required_Age	Win_Flag	Mac_Flag	Linux_Flag	Memory	S
0	NaN	NaN	0	True	False	False	512 MB	
1	52.0	52.0	0	True	True	False	1 GB	
4	0.0	0.0	0	True	True	False	2 GB	
5	0.0	0.0	0	True	False	False	2 GB	
6	530.0	530.0	0	True	False	False	2 GB	
...	...	...	...	...	...	...	...	
80504	299.0	299.0	0	True	False	False	8 GB	
80506	300.0	36.0	0	True	True	False	4 GB	
80507	450.0	360.0	0	True	False	False	6 GB	
80508	1199.0	1199.0	0	True	False	False	4 GB	
80509	480.0	480.0	0	True	False	False	5 GB	

45365 rows × 12 columns

In [ ]:
```
bad_classified["target"] = bad_classified.apply(assign_target, axis=1)
bad_classified
```

```
C:\Users\Lenovo\AppData\Local\Temp\ipykernel_5728\2805256452.py:1: SettingWithCopyWa
rning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/u
ser_guide/indexing.html#returning-a-view-versus-a-copy
  bad_classified["target"] = bad_classified.apply(assign_target, axis=1)
```

Out[ ]:

	Initial_Price	Final_Price	Required_Age	Win_Flag	Mac_Flag	Linux_Flag	Memory	S
3	NaN	NaN	0	True	True	True	2 GB	
7	349.0	349.0	0	True	False	False	1 GB	
12	85.0	85.0	0	True	False	False	NaN	
19	349.0	349.0	0	True	False	False	2 GB	
23	125.0	125.0	0	True	False	False	2 GB	
...	...	...	...	...	...	...	...	
80491	349.0	349.0	0	True	True	False	1 GB	
80498	1300.0	1300.0	0	True	False	False	8 GB	
80499	164.0	164.0	0	True	False	False	4 GB	
80502	610.0	610.0	0	True	False	False	4 GB	
80505	570.0	285.0	0	True	False	False	1 GB	

25362 rows × 12 columns

In [ ]:
```
final_df = pd.concat([good_classified,bad_classified])
final_df
```

Out[ ]:

	Initial_Price	Final_Price	Required_Age	Win_Flag	Mac_Flag	Linux_Flag	Memory	S
0	NaN	NaN	0	True	False	False	512 MB	
1	52.0	52.0	0	True	True	False	1 GB	
4	0.0	0.0	0	True	True	False	2 GB	
5	0.0	0.0	0	True	False	False	2 GB	
6	530.0	530.0	0	True	False	False	2 GB	
...	...	...	...	...	...	...	...	
80491	349.0	349.0	0	True	True	False	1 GB	
80498	1300.0	1300.0	0	True	False	False	8 GB	
80499	164.0	164.0	0	True	False	False	4 GB	
80502	610.0	610.0	0	True	False	False	4 GB	
80505	570.0	285.0	0	True	False	False	1 GB	

70727 rows × 12 columns

Memory and storage process

In [ ]:
```
final_df.dropna(inplace=True,subset=["Memory","Storage"])
```

In [ ]:
```
import re


def extract_number(num : str):
    return float(re.sub(',' , '' , num))

def extract_memory_or_storage(num : str):
    try:
        n,unit = num.split()

        if unit == 'GB':
            return int(n) * 1024
        elif unit == 'MB':
            return int(n)
        return 0
    except:
        print(num)
# Parsing storage and memory

def filter_fn(row):

    if len(row["Memory"].split()) == 2 and len(row["Storage"].split()) == 2:
        return True
    return False

m = final_df.apply(filter_fn, axis=1)
data = final_df[m]
data["Memory_MB"] =  data["Memory"].apply(extract_memory_or_storage)
data["Storage_MB"] =  data["Storage"].apply(extract_memory_or_storage)
data.drop(["Memory" ,"Storage","Overall_Review_Summary"],axis=1,inplace=True)
```

```
C:\Users\Lenovo\AppData\Local\Temp\ipykernel_5728\3669446147.py:28: SettingWithCopyW
arning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/u
ser_guide/indexing.html#returning-a-view-versus-a-copy
  data["Memory_MB"] =  data["Memory"].apply(extract_memory_or_storage)
C:\Users\Lenovo\AppData\Local\Temp\ipykernel_5728\3669446147.py:29: SettingWithCopyW
arning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/u
ser_guide/indexing.html#returning-a-view-versus-a-copy
  data["Storage_MB"] =  data["Storage"].apply(extract_memory_or_storage)
C:\Users\Lenovo\AppData\Local\Temp\ipykernel_5728\3669446147.py:30: SettingWithCopyW
arning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/u
ser_guide/indexing.html#returning-a-view-versus-a-copy
  data.drop(["Memory" ,"Storage","Overall_Review_Summary"],axis=1,inplace=True)
```

In [ ]:
```
# data.to_csv("processed.csv",header=True,index=False)
```

Reviews AND Flags parsing

In [ ]:
```
# data["Positive_Reviews"] = data["Positive_Reviews"].apply(extract_number)
# data["Negative_Reviews"] = data["Negative_Reviews"].apply(extract_number)

data.Win_Flag = data.Win_Flag.astype(bool)
data.Mac_Flag = data.Mac_Flag.astype(bool)
data.Linux_Flag = data.Linux_Flag.astype(bool)
data.Required_Age = data.Required_Age.astype(float)
data.dtypes
```

```
C:\Users\Lenovo\AppData\Local\Temp\ipykernel_5728\1725034132.py:4: SettingWithCopyWa
rning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/u
ser_guide/indexing.html#returning-a-view-versus-a-copy
  data.Win_Flag = data.Win_Flag.astype(bool)
C:\Users\Lenovo\AppData\Local\Temp\ipykernel_5728\1725034132.py:5: SettingWithCopyWa
rning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/u
ser_guide/indexing.html#returning-a-view-versus-a-copy
  data.Mac_Flag = data.Mac_Flag.astype(bool)
C:\Users\Lenovo\AppData\Local\Temp\ipykernel_5728\1725034132.py:6: SettingWithCopyWa
rning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/u
ser_guide/indexing.html#returning-a-view-versus-a-copy
  data.Linux_Flag = data.Linux_Flag.astype(bool)
C:\Users\Lenovo\AppData\Local\Temp\ipykernel_5728\1725034132.py:7: SettingWithCopyWa
rning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/u
ser_guide/indexing.html#returning-a-view-versus-a-copy
  data.Required_Age = data.Required_Age.astype(float)
```

Out[ ]:
```
Initial_Price      float64
Final_Price        float64
Required_Age       float64
Win_Flag              bool
Mac_Flag              bool
Linux_Flag            bool
Positive_Reviews   float64
Negative_Reviews   float64
target               int64
Memory_MB           uint64
Storage_MB          uint64
dtype: object
```

In [ ]:
```
y = data["target"]
X = data.drop(["target"],axis=1)
```

In [ ]:
```
#
# +ve, mixed , -ve
#   1     0     -1
df["Overall_Review_Summary"].unique()
```

Out[ ]:
```
array(['Mostly Negative', 'Very Positive', nan, '5 user reviews', 'Mixed',
       '1 user reviews', 'Mostly Positive', '6 user reviews', 'Positive',
       '3 user reviews', '8 user reviews', 'No user reviews',
       '4 user reviews', 'Negative', 'Overwhelmingly Positive',
       '2 user reviews', '9 user reviews', '7 user reviews',
       'Overwhelmingly Negative', 'Very Negative'], dtype=object)
```