Phase 1 Project: Ebay Perfume Analysis

Authored by Trinity Gahagen

Objective

In this analysis, we will be exploring and analyzing different perfumes sold on ebay to research competition across different criteria.

Dataset used:

https://www.kaggle.com/datasets/kanchana1990/perfume-e-commerce-dataset-2024

External References:

- https://medium.com/@charlesmanimbo/the-difference-between-perfume-cologne-eau-de-toilette-and-more-34ac21281226
- https://www.ebay.com/

Dataset Overview

This dataset consisted of 2 CSV files, one with data pertaining to men's perfumes, and another pertaining to women's. The columns included in each file are:

- `brand` : The brand of the perfume
- `title` : The title of the ebay listing
- `type` : The type of perfume being sold (eau de parfum, eau de toilette, etc.)
- `price` : The price that the perfume is listed as
- `priceWithCurrency` : A text version of the price column, including the currency
- `available` : The number of products available in stock
- `availableText` : The information that ebay provides about availability
- `sold` : The number of products already sold
- `lastUpdated` : The date that the listing was last updated
- `itemLocation` : A list of locations where the item can ship from

The first thing to do is to import the proper libraries and set parameters for the plots.

In [210…

```
# Import necessary modules
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import matplotlib as mpl
```

```
from matplotlib.ticker import FuncFormatter

# Set parameters for plots
plt.style.use('seaborn-v0_8-darkgrid')
plt.rcParams['font.size'] = 14
plt.rcParams['axes.grid'] = True
plt.rcParams['axes.titlesize'] = 16
plt.rcParams['axes.titleweight'] = 'bold'
plt.rcParams['axes.labelweight'] = 'light'

purple_colors = ['#29004a', '#350061', '#3f0073', '#4a0087', '#6200b3',
'#6200b3', '#7d00e3', '#8b00fc', '#9717ff', '#a02bff', '#ac47ff',
'#bc6bff', '#c782ff', '#d29cff', '#daadff', '#e0baff']
red_colors = ['#6F1400', '#6F1400', '#A51D00', '#BB2100', '#D02500',
'#E52800', '#F12B00', '#FF2D00', '#FF3A0F', '#FF4A23', '#FF5834',
'#FF6442', '#FF7051', '#FF8166', '#FF8A71', '#FF937C']
blue_colors = ['#01005B', '#020073', '#020088', '#02009C', '#0300B4',
'#0300CD', '#0300E2', '#0400F5', '#0400FF', '#1713FF', '#2C28FF',
'#403DFF', '#524FFF', '#6361FF', '#7573FF', '#8381FF']
```

Next, we read in the data and view it to get a sense of the formatting and content.

In [211…

```
mens_perfume = pd.read_csv('ebay_mens_perfume.csv')
womens_perfume = pd.read_csv('ebay_womens_perfume.csv')
```

In [212…

```
mens_perfume.head()
```

Out[212…

	brand	title	type	price	priceWithCurrency	available	availableText	sold	lastUpda
0	Dior	Christian Dior Sauvage Men's EDP 3.4 oz Fragra...	Eau de Parfum	84.99	US $84.99/ea	10.0	More than 10 available / 116 sold	116.0	May 2 10:03:04
1	AS SHOW	A-v-entus Eau de Parfum 3.3 oz 100ML Millesime...	Eau de Parfum	109.99	US $109.99	8.0	8 available / 48 sold	48.0	May 2 23:07:49
2	Unbranded	HOGO BOSS cologne For Men 3.4 oz	Eau de Toilette	100.00	US $100.00	10.0	More than 10 available / 27 sold	27.0	May 2 21:55:43

	brand	title	type	price	priceWithCurrency	available	availableText	sold	lastUpda
3	Giorgio Armani	Acqua Di Gio by Giorgio Armani 6.7 Fl oz Eau D...	Eau de Toilette	44.99	US $44.99/ea	2.0	2 available / 159 sold	159.0	May 2 03:30:43
4	Lattafa	Lattafa Men's Hayaati Al Maleky EDP Spray 3.4 ...	Fragrances	16.91	US $16.91	NaN	Limited quantity available / 156 sold	156.0	May 2 07:56:25

In [213…

```
womens_perfume.head()
```

Out[213…

	brand	title	type	price	priceWithCurrency	available	availableText	sold	lastUpdated	it
0	Carolina Herrera	Good Girl by Carolina Herrera 2.7 oz Eau De Pa...	Eau de Parfum	43.99	US $43.99/ea	2.0	2 available / 393 sold	393.0	May 23, 2024 10:43:50 PDT	U
1	As Shown	Parfums de Marly Delina La Rosee Eau de Parfum...	Eau de Parfum	79.99	US $79.99	5.0	5 available / 40 sold	40.0	May 24, 2024 00:15:48 PDT	
2	PRADA	PRADA Paradoxe by Prada EDP 3.0oz/90ml Spray P...	Eau de Parfum	59.99	US $59.99	10.0	More than 10 available / 35 sold	35.0	May 14, 2024 20:54:25 PDT	C Je
3	As Show	J'adore Parfum D'eau by Christian 3.4 oz EDP F...	Eau de Parfum	59.99	US $59.99/ea	10.0	More than 10 available / 9 sold	9.0	May 23, 2024 01:23:05 PDT	.
4	Khadlaj	Shiyaaka for Men EDP Spray 100ML (3.4 FL.OZ) B...	Eau de Parfum	29.99	US $29.99/ea	10.0	More than 10 available	NaN	NaN	U

Data Cleaning

The cells containing code that pertain to cleaning the data, including text processing, null value dropping and imputing, and feature engineering are in the following section.

First, I created columns for each table indicating what sex the perfume is intended for, including those that specify unisex.

In [214…
```
mens_perfume['forMen'] = 1
womens_perfume['forWomen'] = 1
womens_perfume[['unisex','forMen']] = 0
mens_perfume[['unisex', 'forWomen']] = 0


print(mens_perfume.columns)
print(womens_perfume.columns)
```

```
Index(['brand', 'title', 'type', 'price', 'priceWithCurrency', 'available',
       'availableText', 'sold', 'lastUpdated', 'itemLocation', 'forMen',
       'unisex', 'forWomen'],
      dtype='object')
Index(['brand', 'title', 'type', 'price', 'priceWithCurrency', 'available',
       'availableText', 'sold', 'lastUpdated', 'itemLocation', 'forWomen',
       'unisex', 'forMen'],
      dtype='object')
```

Then I combined these two tables by concatenating them to make one singular DataFrame.

In [215…
```
perfumes = pd.concat([mens_perfume, womens_perfume], ignore_index=True)
```

In [216…
```
perfumes.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2000 entries, 0 to 1999
Data columns (total 13 columns):
 #   Column             Non-Null Count  Dtype  
---  ------             --------------  -----  
 0   brand              1998 non-null   object 
 1   title              2000 non-null   object 
 2   type               1995 non-null   object 
 3   price              2000 non-null   float64
 4   priceWithCurrency  2000 non-null   object 
 5   available          1758 non-null   float64
 6   availableText      1989 non-null   object 
 7   sold               1978 non-null   float64
 8   lastUpdated        1874 non-null   object 
 9   itemLocation       2000 non-null   object 
 10  forMen             2000 non-null   int64  
 11  unisex             2000 non-null   int64  
 12  forWomen           2000 non-null   int64  
dtypes: float64(3), int64(3), object(7)
memory usage: 203.2+ KB
```

There are very few null values based off brand and type, so dropping them will be inconsequential.

In [217…
```
perfumes.dropna(subset=['brand', 'type'], inplace=True)
```

Here I created columns to indicate whether the perfume was intended for men, women, or unisex.

In [218…
```
# Create masks to filter for each gender
UNISEX_MASK = perfumes['title'].str.lower().str.contains('unisex')
WOMEN_MASK = perfumes['title'].str.lower().str.contains('for women')
MEN_MASK = perfumes['title'].str.lower().str.contains('for men')

# Filter using np.where(), changing the values where the condition is true,
otherwise leave it as it is
perfumes['unisex'] = np.where(UNISEX_MASK, 1, perfumes['unisex'])
perfumes['forWomen'] = np.where(WOMEN_MASK, 1, perfumes['forWomen'])
perfumes['forMen'] = np.where(MEN_MASK, 1, perfumes['forMen'])

# Make sure that there are no rows with 1s in more than one of these
columns
for col in ['forWomen', 'forMen']:
    perfumes[col] = np.where(perfumes['unisex'] == 1, 0, perfumes[col])
```

Text Preprocessing

The following section of code was preprocessing the text in the `type`, `location`, and `brand` columms to make each of them more consistently formatted. One hot encoding was used to indicate location from which each perfume shipped.

In [219…
```
perfumes['type'].unique()
```

Out[219…
```
array(['Eau de Parfum', 'Eau de Toilette', 'Fragrances', 'Perfume', '/',
       'PARFUM', 'Parfum', 'Concentrated Uncut Pure Body Oil',
       'LE PARFUM', 'Eau De Parfum', 'Unscented', 'Eau de Cologne',
       '~ THE ONE EAU DE PARFUM SPRAY ~', 'EXTRAIT DE PARFUM',
       'Eau De Toilette', 'Eau De Parfum Intense', 'Pheromone',
       'Aftershave', 'Fragrance & Perfume', 'Eau de Perfume',
       'Jo Malone Cologne Intense Spray', 'Y', 'Gift Sets',
       'Fragrance Rolling Ball', 'Body Spray', 'Eau de toilette',
       'Eau de Toillette',
       'Eau De Toilette, Eau De Parfum, Eau De Parfum Intense', 'Cologne',
       'le parfum', 'Eau de Toilette Intense',
       'Eau de Cologne Spray, Cologne Spray', 'Extrait De Parfum',
       'Fine Cologne', 'Does not apply', 'EDT', 'Extrait de Parfum',
       'Editions Parfums', 'DIOR HOMME COLOGNE', 'Deodorant', 'De Nuit',
       'Eau de Toilette, Cologne Spray', 'Parfum Intense',
       'Eau de Parfum Intense', 'cologne', 'EAU DE COLOGNE SPRAY',
       'Roll On', 'Elixir de Parfum', 'Elixir', 'Various', 'Assorted',
       'Fragrance Oil', 'Eau de Parfum/ Eau de Toilette',
       'Deodorant Body Spray', 'Oil', 'Splash-on', 'edt', 'EDC',
       'Car Air Freshener', 'Eau De Cologne', 'Fragrance Body Spray',
       'Body Oil', 'Cologne spray', 'Spray',
       '~ BODY FIRM ADVANCED BODY REPAIR TREATMENT ~', 'Fragrance Mist',
       'Deodorant Stick', '3 Pc', 'Eau de Parfum, Eau De Parfume', 'Mist',
       'Eau de Toilette, Spray', 'Perfume, Eau de Parfum', 'Hair Perfume',
       'Eau de Parfum, Spray', 'Cream', 'Eau De Parfum 2 Pcs Set',
       "L'Eau de Parfum", 'Eau de Parfume', "L'eau de Parfum",
       'SKIN_MOISTURIZER', 'EDP and Parfum', 'Fine Fragrance Mist',
       'Lotion', 'Scented Oils', 'Shimmer', 'EDP',
       'Perfume Fragrance Mist', 'Perfume Oil',
       'Parfum, Lotion, Gloss and Blush', 'Bath Oil', 'Elixir De Parfum',
       'Body Mist', 'ASST', 'BEAUTY', 'Extract Parfum', 'Does Not Apply',
       'Shimmering Body Oil', 'Body Powder', 'Eau De Parfum Supreme',
       'Perfume Gift Sets', 'Toilette Spray',
       'Eau De Parfum Spray (Unisex Tester) 3.4 oz', 'Body Lotion',
       'deodorant', 'Esprit de Parfum', 'Discovery Set',
       'SOLID PERFUME STICK', 'Eau De Toilette Spray 3.4 oz',
       'Eau De Parfum 3 Pcs Set', 'Sensuous Body Moisturizer',
       'Cologne Spray', 'Pink Sugar', 'Oil Perfume',
       'FRAGRANCE BODY MIST', 'Eau de Parfum/Perfume', '1'], dtype=object)
```

In [220…

```python
# Find phrases or words to match on
eau_de_toilette = "(eau){1} de toi.*|edt"
eau_de_parfum = "(eau){1} de parfum|edp|eau de perfume"
mist = "mist"
cologne = "cologne|edc"
oil = "(?!t)oil"
fragrance = "fragrance"
roll_on = "roll"
lotion = "lotion|moisturizer|cream"
deodorant = "deodorant"
sets = "set|pc"
perfume = "perfume|^parfum"

# Put them in a list
type_masks = [eau_de_toilette, eau_de_parfum, mist, cologne, oil,
fragrance, roll_on, lotion, deodorant, sets, perfume]

# Create labels
type_categories = [
    "Eau De Toilette",
    "Eau De Parfum",
    "Mist",
    "Cologne",
    "Oil",
    "Fragrance",
    "Roll On",
    "Lotion",
    "Deodorant",
    "Sets",
    "Perfume"
]


# Create a list of conditions to filter and replace on
type_conditions = [(perfumes['type'].str.lower().str.contains(condition,
na=False)) for condition in type_masks]
```

```
c:\Users\twitt\anaconda3\envs\learn-env\lib\site-packages\pandas\core\strings.py:2001: U
serWarning: This pattern has match groups. To actually get the groups, use str.extract.
  return func(self, *args, **kwargs)
```

In [221…

```
# Use np.select() in a similar fashion to np.where(), but this time just
place all the other categories in "Other"
perfumes['type_clean'] = np.select(type_conditions, type_categories,
default="Other")
```

In [222…

```
print(perfumes['type_clean'].unique())
```

```
['Eau De Parfum' 'Eau De Toilette' 'Fragrance' 'Perfume' 'Other' 'Oil'
 'Cologne' 'Sets' 'Deodorant' 'Roll On' 'Mist' 'Lotion']
```

In [223…

```
# Rinse and repeat with location
perfumes['itemLocation'].unique()
```

Out[223…

```
array(['Allen Park, Michigan, United States', 'Atlanta, Georgia, Canada',
       'Dearborn, Michigan, United States',
       'Reinholds, Pennsylvania, United States',
       'Brooklyn, New York, United States',
       'Houston, Texas, United States',
       'Englewood Cliffs, New Jersey, United States',
       'Ithaca, New York, United States', 'shanghai, China',
       'Dearborn Heights, Michigan, United States',
       'Ecorse, Michigan, United States',
       'Warren, Michigan, United States',
       'San Francisco, California, United States',
       'Dayton,New Jersey, Hong Kong',
       'San Jose, California, United States',
       'Miami, Florida, United States',
       'Hamtramck, Michigan, United States',
       'Flat Lick, Kentucky, United States',
       'Elmhurst, New York, United States',
       'Hackensack, New Jersey, United States',
       'Dallas, Texas, United States',
       'Pomona, California, United States', 'Katy, Texas, United States',
       'College Point, New York, United States',
       'NEW YORK, United States', 'New York,USA, Hong Kong',
       'Brenham, Texas, United States',
       'Richmond Hill, New York, United States',
       'Dexter, Michigan, United States',
       'Phoenix, Arizona, United States',
       'Peekskill, New York, United States',
       'Edison, New Jersey, United States',
       'Armada, Michigan, United States',
       'Kalamazoo, Michigan, United States',
       'Charlotte, North Carolina, United States',
       'Dayton, New Jersey, United States',
       'Perth Amboy, New Jersey, United States',
       'LaGrange, Georgia, United States',
       'Cranston, Rhode Island, United States',
       'Las Vegas, Nevada, United States',
       'Sacramento, California, United States',
       'Dayton, New Jersey, Canada',
       'Windsor Mill, Maryland, United States',
       'Mesa, Arizona, United States',
       'West Orange, New Jersey, United States',
       'Henrico, Virginia, United States',
       'Pfafftown, North Carolina, United States', 'CA, United States',
       'USA,California, Hong Kong', 'New York, New York, Taiwan',
       'Orange, Connecticut, United States',
       'Hephzibah, Georgia, United States',
       'El Cajon, California, Hong Kong',
       'California,United States, Hong Kong, Hong Kong',
       'Fresh Meadows, New York, United States',
       'Cumming, Georgia, United States',
       'Detroit, Michigan, United States',
       'Walnut, California, Hong Kong', 'Dayton, New Jersey, Hong Kong',
       'Ottawa,California, Canada',
       'Highland Park, Michigan, United States',
       'Melissa, Texas, United States',
       'Carlsbad, California, United States',
       'Livingston Manor, New York, United States',
       'Hollywood, Florida, United States',
       'Laughlin, Nevada, United States',
       'Albany, New York, United States',
       'New York, New York, United States', 'Hong Kong, Hong Kong',
       'Austin, Texas, United States',
       'Los Angeles, California, United States',
       'Lubbock, Texas, United States',
       'Jonesboro, Georgia, United States',
       'Englewood, New Jersey, United States',
       'Philadelphia, Pennsylvania, United States',
       'Santa Cruz, California, United States',
       'Baltimore, Maryland, United States', 'HongKong, Hong Kong',
       'Webster, Florida, United States', 'Shenzhen, China',
       'Melvindale, Michigan, United States',
       'Thomasville, Alabama, United States',
       'Ontario, California, United States, Hong Kong',
       'Hialeah, Florida, United States',
       'Chesapeake, Virginia, United States',
       'Farmington, Michigan, United States',
       'Des Moines, Iowa, United States',
       'Grand Rapids, Michigan, United States',
       'Harwood Heights, Illinois, United States',
       'Macomb, Michigan, United States', 'New Jersey, United States',
       'Pearland, Texas, United States', 'California USA Taiwan, Taiwan',
       'New City, New York, United States',
       'Cincinnati, Ohio, United States',
       'Melbourne, Florida, United States',
       'Houston, Texas, HongKong, China',
       'Bethpage, New York, United States',
       'Phillipsburg, New Jersey, United States',
       'Hongkong/California, Hong Kong', 'Toronto,Ontario, Canada',
       'Southgate, Michigan, United States',
       'Bogota, New Jersey, United States',
       'Euless, Texas, United States', 'New York, Hong Kong',
       '27704, United States', 'New York,United States, Hong Kong',
       'North Brunswick, New Jersey, United States',
       'Lynwood, California, United States',
       'Woodbridge, Virginia, United States',
       'Sterling Heights, Michigan, United States',
       'California, United States, Hong Kong',
       'Stafford, Texas, United States', 'Houston, Texas,HongKong, China',
       'Falls Church, Virginia, United States',
       'Fredericksburg, Virginia, United States',
       'Oneida, Tennessee, United States',
       'Egg Harbor Township, New Jersey, United States',
       'Midway City, California, United States',
       'San Gabriel, California, United States', 'TX, United States',
       'Taiwan, Taiwan', 'Rochester, New York, United States',
       'Glendora, California, United States',
       'California or Hong Kong, Hong Kong',
       'Orlando, Florida, United States', 'Toronto, Ontario, Canada',
       'Scarborough, Ontario, Canada', 'Austell, Georgia, United States',
       'Winter Garden, Florida, United States',
       'San Diego, California, United States',
       'Hong Kong or Virginia, Hong Kong',
       'Ronkonkoma, New York, United States',
       'Buhl, Idaho, United States', 'California, Hong Kong',
       'Lake Zurich, Illinois, United States',
       'Dunedin, Florida, United States',
       'Winter Haven, Florida, United States',
       'Modesto, California, United States',
       'Seattle, Washington, United States', 'HK, Hong Kong',
       'Salem, Virginia, United States',
       'Nevada/California, USA, Hong Kong, Hong Kong',
       'Westminster, California, United States', 'Orlando, United States',
       'ShenZhen, China', 'Midway City, California, Hong Kong',
       'Graham, Washington, United States',
       'South San Francisco, California, United States',
       'Spring Hill, Florida, United States',
       'Fort Lauderdale, Florida, United States',
       'Schenectady, New York, United States',
       'Rowland Heights, California, United States',
       'Princeton, New Jersey, United States',
       'Fenton, Michigan, United States',
       'Centralia, Washington, United States', 'NewYork, Hong Kong',
       ' Beauty Maxima, United States, United States',
       'Nashua, New Hampshire, United States',
       'Union, New Jersey, United States',
       'Franklin Square, New York, United States',
       'Sterrett, Alabama, United States',
       'El Cajon, California, United States',
       'Romulus, Michigan, United States',
       'Bridgeview, Illinois, United States',
       'Lincoln Park, Michigan, United States',
       'Fourmile, Kentucky, United States',
       'Hicksville, New York, United States',
       'Virginia Beach, Virginia, United States',
       'New York,United States, Taiwan',
       'Sammamish, Washington, United States',
       'Princeton Junction, New Jersey, United States',
       'Phenix City, Alabama, United States',
       'Dayton,New Jersey,USA, Hong Kong',
       'McLean, Virginia, United States',
       'Memphis, Tennessee, United States',
       'Multiple Location, United States',
       'Jersey City, New Jersey, United States',
       'Sanborn, New York, United States',
       'Cabot, Arkansas, United States', 'Dover, Delaware, United States',
       'Debary, Florida, United States', 'United States, Hong Kong',
       'Fort Walton Beach, Florida, United States',
       'Jacksonville, Florida, United States',
       'North York, Ontario, Canada', 'United States',
       'Brownsboro, Texas, United States',
       'Jamaica, New York, United States',
       'Keyport, New Jersey, United States', 'Ottawa, Canada',
       'Texas, Hong Kong', 'Green Bay, Wisconsin, United States',
       'Columbia, South Carolina, United States',
       'Matawan, New Jersey, United States',
       'Maiden, North Carolina, United States',
       'Towson, Maryland, United States', 'United states, United States',
       'Monterey Park, California, United States',
       'Douglasville, Georgia, United States', 'Jerusalem, Israel',
       'Lakewood, California, United States', 'Elk, Poland',
       'WA, United States', 'Gonzales, Louisiana, United States',
       'Hong Kong, China', 'Montclair, California, United States',
       'Inkster, Michigan, United States',
       'Huntsville, Alabama, United States',
       'Valley Stream, New York, United States',
       'Long Beach, California, United States',
       'Middletown, Delaware, United States', 'New York, United States',
       '36784, United States', 'California,United States, Hong Kong',
       'Appleton, Wisconsin, United States',
       'United States,California, Taiwan',
       'Rockledge, Florida, United States',
       'Belleville, Michigan, United States',
       'Flemington, New Jersey, United States',
       'Orange City, Florida, United States', 'REGENTE FEIJO, Brazil',
       'Madera, California, United States',
       'West Harrison, Indiana, United States',
       'Grafton, Wisconsin, United States',
       'Wakefield, Massachusetts, Hong Kong',
       'Hamlet, North Carolina, United States', 'New Jersey, Hong Kong',
       'Wynnewood, Pennsylvania, United States',
       'Monroe Township, New Jersey, Hong Kong',
       'St. George, Utah, United States',
       'East Meadow, New York, United States',
       'California,USA, Hong Kong', 'Humacao, Puerto Rico, United States',
       'Huntington Station, New York, United States',
       'Saint Clair Shores, Michigan, United States',
       'Middle Grove, New York, United States',
       'Temple, Texas, United States',
       'Nine Mile Falls, Washington, United States',
       'Matosinhos, Portugal', 'Whittier, California, United States',
       'Plano, Texas, United States',
       'Pompano Beach, Florida, United States',
       'Shepherdsville, Kentucky, United States',
       'Irving, Texas, United States',
       'Orma, West Virginia, United States',
       'Pomona,California, Hong Kong', '20744, United States',
       'West Chester, Pennsylvania, United States', 'DELHI, DELHI, India',
       'Corona, California, United States',
       'Punta Gorda, Florida, United States',
       'Saint Petersburg, Florida, United States',
       'Arcadia, California, United States',
       'Roseville, California, United States',
       'Lynnwood, Washington, United States',
       'Ann Arbor, Michigan, United States',
       'Paterson, New Jersey, United States',
       'Battle Creek, Michigan, United States',
       'Edison, New Jersey, Hong Kong',
       'Boynton Beach, Florida, United States',
       'California,United States,HK, Hong Kong',
       'Port Orange, Florida, United States',
       'Ontario, California, United States',
       'Lynbrook, New York, United States',
       'Stony Brook, New York, United States',
       'Multiple Locations, United States',
       'Arlington Heights, Illinois, United States',
       'Astoria, New York, United States', 'Burlington, Ontario, Canada',
       'Selinsgrove, Pennsylvania, United States',
       'Canton, Michigan, United States', 'Argyle, Texas, United States',
       'Bellaire, Texas, United States', 'Manor, Texas, United States',
       'Chico, California, United States',
       'Orange, New Jersey, United States', 'USA, New Jersey, Hong Kong',
       'Little Ferry, New Jersey, United States',
       'Quinlan, Texas, United States',
       'Oklahoma City, Oklahoma, United States',
       'Melrose Park, Illinois, United States',
       'Stuyvesant Falls, New York, United States',
       'Portland, Oregon, United States',
       'South El Monte, California, United States',
       'United States, Canada', 'CA, China',
       'Round Rock, Texas, United States',
       'Tulsa, Oklahoma, United States',
       'Elizabeth, New Jersey, United States',
       'Buffalo Mills, Pennsylvania, United States',
       'El Monte, California, United States', 'Plano, United States',
       'Kennesaw, Georgia, United States',
       'Van Nuys, California, United States',
       'Richmond, Texas, United States',
       'Walled Lake, Michigan, United States',
       'New Jersey,United States, Hong Kong',
       'Lisle, Illinois, United States',
       'Durham, North Carolina, United States',
       'Bowling Green, Kentucky, United States',
       'Beverly Hills, California, United States',
       'Cranford, New Jersey, United States',
       'Sayreville, New Jersey, United States',
       'Flushing, New York, United States',
       'Mogadore, Ohio, United States',
       'Rocklin, California, United States',
       'Oxnard, California, United States',
       'Nashville, Tennessee, United States',
       'Elgin, Illinois, United States',
       'Bluff City, Tennessee, United States',
       'Clearwater, Florida, United States',
       'Wytheville, Virginia, United States',
       'Chicago, Illinois, United States',
       'Potomac, Maryland, United States',
       'Cayuga, New York, United States',
       'Villa Rica, Georgia, United States',
       'Okemos, Michigan, United States',
       'Sea Cliff, New York, United States', 'New York, USA, Hong Kong',
       'Alexandria, Minnesota, United States',
       'Long Island City, New York, United States',
       'Montgomery, Alabama, United States',
       'Tallman, New York, United States', 'CA.San Francisco, Hong Kong',
       ' SAN FRANCISCO, CA, USA, Hong Kong',
       'Piscataway, New Jersey, United States',
       'Troy, New York, United States',
       'Amigo, West Virginia, United States',
       'Giddings, Texas, United States',
       'California USA Hongkong, Hong Kong',
       'Laguna Niguel, California, United States',
       'Florence, South Carolina, United States',
       'Ashburnham, Massachusetts, United States',
       'Plainfield, Illinois, United States',
       'West Palm Beach, Florida, United States',
       'Woodruff, South Carolina, United States',
       'Katy, Texas, Hong Kong', 'Newark, Delaware, United States',
       'Syracuse, New York, United States',
       'Kew Gardens, New York, United States',
       'Jefferson, Texas, United States',
       'Joppa, Maryland, United States',
       'Bakersfield, California, United States',
       'Tarzana, California, United States',
       'Warminster, Pennsylvania, United States',
       'Linden, New Jersey, United States',
       'Kansas City, Missouri, United States', 'New York,USA, Taiwan',
       'California, United States', 'Rockville, Maryland, United States',
       'Camarillo, California, United States',
       'Novi, Michigan, United States',
       'Old Bethpage, New York, United States',
       'Los Angeles,California, United States',
       'Liberty, Texas, United States',
       'Arlington, Virginia, United States',
       'Columbus, Georgia, United States',
       'New York, United States, Hong Kong',
       'Glendale, California, United States', '89101, United States',
       'Fort Wayne, Indiana, United States',
       'Niagara Falls, New York, United States',
       'Hamptonville, North Carolina, United States',
       'Rowlett, Texas, United States',
       'Atlantic Beach, Florida, United States',
       'Townsend, Massachusetts, United States',
       'Oceanside, New York, United States',
       'Miles City, Montana, United States',
       'Layton, Utah, United States',
       'Saint Clair, Michigan, United States',
       'Congers, New York, United States',
       'Citrus Heights, California, United States',
       'McAdenville, North Carolina, United States',
       'Staten Island, New York, United States',
       'Boiling Springs, Pennsylvania, United States',
       'Maryville, Tennessee, United States',
       'Gobles, Michigan, United States',
       'Colorado Springs, Colorado, United States',
       'Frederick, Maryland, United States',
       'Branford, Connecticut, United States',
       'Huntsville, Texas, United States',
       'Elberton, Georgia, United States',
       'New Jersey,United Stated, Hong Kong',
       'Minneapolis, Minnesota, United States',
       'Chatsworth, California, United States',
       'Chantilly, Virginia, United States',
       'Somerset, Pennsylvania, United States',
       'Rock Rapids, Iowa, United States',
       'Henderson, Nevada, United States',
       'Louisville, Kentucky, United States',
       'East Brunswick, New Jersey, United States',
       'Pomona,California USA, Hong Kong', 'Payson, Utah, United States',
       'Pembroke, North Carolina, United States',
       'Anaheim, California, United States',
       'Forest Hills, New York, United States',
       'Sarasota, Florida, United States',
       'San Marcos, California, United States',
       'Point Pleasant Beach, New Jersey, United States',
       'Hauppauge, New York, United States',
       'Fountain Valley, California, United States',
       'Running Springs, California, United States',
       'Pawtucket, Rhode Island, United States',
       'Santa Ana, California, United States',
       'Manchester, New Hampshire, United States',
       'Kissimmee, Florida, United States',
       'Tucson, Arizona, United States',
       'Feasterville-Trevose, Pennsylvania, United States',
       'Takamatsu City, Japan', 'Saint Augustine, Florida, United States',
       'Weatherford, Texas, United States',
       'Redmond, Washington, United States',
       'La Porte, Indiana, United States',
       'Stanley, Wisconsin, United States', '44260, United States',
       'Mountain Top, Pennsylvania, United States',
       'Palm Desert, California, United States',
       'Addison, Illinois, United States',
       'New Hyde Park, New York, United States',
       'Sayreville, New Jersey, Pakistan',
       'West Hollywood, California, United States',
       'Temecula, California, United States', 'Sevlievo, Bulgaria',
       'Spring Valley, New York, United States',
       'Myrtle Beach, South Carolina, United States',
       'Eden Prairie, Minnesota, United States',
       'Carlton, Georgia, United States',
       'Irvine, California, United States',
       'Hemet, California, United States', 'Paige, Texas, United States',
       'Bainbridge Island, Washington, United States',
       'Sunnyside, New York, United States',
       'Lake Hiawatha, New Jersey, United States',
       'Lincoln, Nebraska, United States',
       'Denham Springs, Louisiana, United States',
       'Deer Park, New York, United States',
       'Houston,Texas,HongKong, China',
       'Pittsburgh, Pennsylvania, United States',
       'South Ozone Park, New York, United States',
       'Burlington, Iowa, United States',
       'Chino, California, United States',
       'Mount Holly, New Jersey, United States',
       'Lodi, California, United States',
       'Poway, California, United States'], dtype=object)
```

In [224…

```
usa = "usa|united states|us|estados unidos|unitedstates"
hong_kong = "hong kong|hk|hongkong"
china = "china"
india = "india"
pakistan = "pakistan"
canada = "canada"
taiwan = "taiwan"
brazil = "brazil"
japan = "japan"
bulgaria = "bulgaria"

location_masks = [usa, hong_kong, china, india, pakistan, canada, taiwan,
brazil, japan, bulgaria]

location_categories = [
    "USA",
    "HK",
    "China",
    "India",
    "Pakistan",
    "Canada",
    "Taiwan",
    "Brazil",
    "Japan",
    "Bulgaria"
]

location_conditions =
[(perfumes['itemLocation'].str.lower().str.contains(condition, na=False))
for condition in location_masks]

zipped_locations = zip(location_categories, location_conditions)
```

In [225…

```
# One hot encode location columns
for place, condition in zipped_locations:
    col_label = f'shipsFrom{place}'
    perfumes[col_label] = condition.map(lambda x: 1 if x == True else 0)
```

In [226…

```
# Finally do the same thing for brand
perfumes['brand'] = perfumes['brand'].str.title().str.strip(' ~')
perfumes['brand'].unique()
```

Out[226…

```
array(['Dior', 'As Show', 'Unbranded', 'Giorgio Armani', 'Lattafa',
       'Multiple Brands', 'Maison Alhambra', 'Gucci', 'Ralph Lauren',
       'Dolce&Gabbana', 'Secertmu', 'Versace', 'Paco Rabanne', 'Grandeur',
       'Armaf', 'Carolina Herrera', 'Dolce & Gabbana', 'Clinique',
       'Dumont', 'Afnan', 'Azzaro', "Penhaligon'S", 'Bharara',
       'Valentino', 'Guy Laroche', 'Montblanc', 'Rasasi', 'Calvin Klein',
       'Uomo', 'Givenchy', 'Polo Ralph Lauren', 'C.K', 'John Varvatos',
       'Nautica', 'As Picture Show', 'Kenneth Cole', 'Tommy Hilfiger',
       '2Nd To None', 'Yves Saint Laurent', 'Cologne', 'As Shown', 'Roja',
       'Metaherbal Labs', 'Mirage Brands', 'Abercrombie & Fitch',
       'Moschino', 'Superz Budapest', 'Gianni Versace', 'Christian Dior',
       'Hermès', 'Diesel', 'Lacoste', 'Dossier', 'Burberry',
       'Michael Malul', 'Zara', 'Aramis', 'Jean Paul Gaultier',
       'Davidoff', 'As Picture Shown', 'Bvlgari', 'Parfums De Marly',
       'Salvatore Ferragamo', 'Ard Al Zaafaran', 'Karl Lagerfeld',
       'J. Del Pozo', 'Sean John', 'Ysl', 'Jaguar', 'Ebc',
       'Bath & Body Works', 'Issey Miyake', 'King', 'Prada', 'Hugo Boss',
       'Mont Blanc', 'Tommy Bahama', 'Paul Sebastian', 'Halloween',
       'Boucheron', 'Thierry Mugler', 'Jo Malone', 'Khadlaj',
       'Louis Vuitton', 'Creed', 'Mfk', 'Hermes', 'Fragrance', 'Chanel',
       'Lalique', 'Liz Claiborne', 'By Al Hambra', 'Joop', 'Ted Lapidus',
       'Axe', 'As Showed', 'Lomani', 'King Of Kings', 'Rue21',
       'Roja Dove', 'Tom Ford', 'Al Wataniah', 'Macarena', 'Bentley',
       'Coach', 'Coty', 'Lanvin', 'Claude Marsal', 'Nikos', 'Lucianno',
       'Viktor & Rolf', 'Rochas', 'Classic Brands', 'Reyane Tradition',
       'Giorgio Beverly Hills', 'Myrurgia', 'Jovan', 'Coty Inc.',
       'Mont Blanc Legend', 'Ulric De Varens', 'Emporio Armani',
       'Maison Francis Kurkdjian', 'Fragrance World', 'Allsaints', 'Avon',
       'Yves De Sistelle', 'Limited Edition', 'Kenneth Cole Reaction',
       'Bond No. 9', 'Acqua Di Parma', 'As Photos', 'Guerlain Paris',
       'Al Haramain', 'Roja Parfums', 'Narciso Rodriguez', 'Topshelf',
       'Brut', 'Ed Hardy', 'Hollister', 'Fm', 'Paris Hilton',
       'Heaven Scents', 'Milestone Perfumes', 'Kenzo', 'Mercedes-Benz',
       'Franck Olivier', 'Missoni', 'Halston', 'Baron', 'Adidas',
       'Guerlain', 'Branded', 'Acqua Di Gio', 'Old Spice',
       'Lauren Ralph Lauren', 'Clive Christian', 'Giorgio Arm.Ani',
       'Victor & Rolf', 'Polo', 'Roberto Cavalli', 'Emanuelle Ungaro',
       'Curve', 'Michael Malul London', 'Estée Lauder', 'Pierre Cardin',
       'Rawchemistry', 'Sterling', 'Territoire', 'Jimmy Choo', 'Lapidus',
       'Mary Kay', 'L’Occitane', 'Have A Scent', 'Fragrance Couture',
       'Mancera', 'Llure Sx', 'Jacques Bogart', 'Fc', 'Pheromones',
       'Cartier', 'Assorted', 'Hinode', 'Michael Jordan',
       'Designer Series', 'Alexandria Fragrances',
       'Maison Martin Margiela', 'El Ganso', 'Victor Manuelle',
       'English Laundry', 'Luxury', 'Jil Sander', 'Hybrid & Company',
       'Perry Ellis', 'Houbigant', 'Does Not Apply', 'Tiffany', 'Emper',
       'Montale', 'Blue Perfumes', 'Fragance One', 'Elixir',
       'Michael Malul Gents Scents', 'Xerjoff', 'United Scents',
       'Aladdin', 'Hanae Mori', 'Phillips-Van Heusen', 'Elizabeth Taylor',
       'Al Rehab', 'Bod Man', 'Amouage', 'Dana', 'Dvyne Fragrances',
       'Arabian Oud', 'David Beckham', 'Falic Fashion Group',
       'Michel Germain', 'The Baron', 'Nina Ricci', 'Vince Camuto',
       'Maison Margiela', 'Mercedes Benz', 'Juicy Couture', 'Cuba',
       'Mandarin Duck', 'Guess', 'Estee Lauder', 'Philosophy',
       'Marc Jacobs', 'Qrc', 'Crepe Erase', 'Ex Nihilo',
       'Juliette Has A Gun', 'Giorgi^O Armani', 'Ariana Grande',
       'Sol De Janeiro', 'Kilian', 'Donna Karan', 'Giorgio² Armani',
       'Elizabeth Arden', 'Urban Outfitters', 'Alt Fragrances',
       "Victoria'S Secret", 'Kim Kardashian', 'Gap', 'Cacharel',
       'Jessica Mcclintock', 'As  Shown', 'Alfred Sung',
       'Gloria Vanderbilt', 'Parfums', 'Ouai', 'Lancôme',
       'Ellis Brooklyn', 'Generic', 'Byredo', 'Huda Beauty Kayali',
       'Evyan', 'Lake&Skye', 'Frederic Malle', 'Marilyn Miglin',
       'Aquolina', 'Lattafa Perfumes', 'Katy Perry', 'Atelier Cologne',
       'Christian Audigier', 'Vera Wang', 'Clean', 'Britney Spears',
       'Escada', 'Michael Kors', 'Le Labo', 'Glossier', 'Dolce Gabbana',
       'Chloe', 'Lancome', 'Natalie', 'Chaka By Chaka Khan',
       'Huda Beauty', 'Bellie Eilish', 'Lucky', 'Tf', 'Oscar De La Renta',
       'Elizabeth And James Nirvana', 'Flower', 'Parfums Gres',
       'Vicki Tiel', 'Fragonard', 'Kate Spade New York', 'Spiritual Sky',
       'Al Hambra', 'True Religion', 'Taylor Swift', 'M·A·C',
       'Robert Piguet', 'Gloss Moderne', 'Justin Bieber', 'Revlon',
       'Sarah Jessica Parker', 'Jennifer Lopez', 'Kate Spade', 'J Lo',
       'Yardley London', 'Pure Romance', 'Tous', "Perfumer'S Workshop",
       'Parfum', 'Adam Levine', 'Jean Couturier', 'Chopard',
       'Change For Women', 'Asst', 'Al-Rehab', 'Tiziana Terenzi',
       'Tory Burch', 'Miraclelayer', 'One Direction',
       'Prince Matchabelli', 'Giardini Di Toscana', 'Jessica Simpson',
       'Love Luxe Beauty', 'Tru Fragrance', 'Sabrina Carpenter', 'Nest',
       'Parfums Grès', 'Paloma Picasso', 'Aeropostale', 'Phlur', 'Jlo',
       'Ainash Perfums', 'Diptyque', 'T.O.V.A.', 'Benetton',
       'Tiffany & Co.', 'Fred Hayman', 'Miim Miic', 'Aerin', 'Pleasing',
       'Pink', 'Perfume', 'Ulta', 'Rihanna', 'Vilhelm Parfumerie',
       'Preferred Fragrance', 'Sisley', 'Anna Sui',
       'Fuller Armand Dupree', 'Atlantic', "L'Artisan", 'Clubman Pinaud',
       'Banana Republic', 'Lolita Lempicka', 'Baby Phat', 'Mem',
       'Gwen Stefani', 'Chloé', 'Jeanne Arthes', 'Daddy Yankee',
       'Classic Erotica', 'Pacifica', 'Snif', 'The Body Shop',
       'Five Star Fragrance', 'Pink Sugar', 'Elizabeth & James',
       'Elie Saab', 'See Photo', 'Armand Dupree Fuller', 'Dkny',
       'United Colors Of Benetton', 'Maison'], dtype=object)
```

In [227…

```
unbranded = "unbranded|.*show.*|.*photo.*|ass[.*]t|change for
women|perfume|does not apply|branded|generic|multiple brands|classic brand"
armani = "giorgio|armani"
calvin_klein = "calvin klein|c[\s\S]k"
dolce = "dolce|gab+an+a"
viktor_rolf = "vi[ck]tor[\s+\S+]rolf"
jlo = "jlo|j lo"
eilish = "eilish"
roja = "roja"
ralph_lauren = "(lauren)\Z|polo"
dior = "dior"
lancome = "lanc.me"
parfums_gres = "parfums gr.s"
al_hambra = "al hambra"
maison_alhambra = "maison alhambra"
mfk = "maison francis kurkdjian|mfk|maison$"
maison_margiela = "margiela"
tiffany = "tiffany[\s\S]co"
versace = "versace"
mercedes = "mercedes"
michael_malul = "malul"
fragrance_one = "^fragance"

brand_masks = [unbranded,
                armani,
                calvin_klein,
                dolce,
                viktor_rolf,
                jlo,
                eilish,
                roja,
                ralph_lauren,
                dior,
                lancome,
                parfums_gres,
                al_hambra,
                maison_alhambra,
                mfk,
                  maison_margiela,
                  tiffany,
                  versace,
                  mercedes,
                  michael_malul,
                  fragrance_one]

brand_categories = [
    "Unbranded",
    "Giorgio Armani",
    "Calvin Klein",
    "Dolce & Gabbana",
    "Viktor & Rolf",
    "J Lo",
    "Billie Eilish",
    "Roja",
    "Ralph Lauren",
    "Dior",
    "Lancome",
    "Parfums Gres",
    "Al Hambra",
    "Maison Alhambra",
    "MFK",
    "Maison Margiela",
    "Tiffany & Co\.",
    "Versace",
    "Mercedes Benz",
    "Michael Malul",
    "Fragrance One"
]



brand_conditions = [(perfumes['brand'].str.lower().str.contains(condition, na=False)) for condition in brand_masks]

zipped = zip(brand_conditions, brand_categories)
```

In [228…

```
for condition, label in zipped:
    perfumes['brand'] = np.where(condition, label, perfumes['brand'])
```

In [229…
```
perfumes['brand'].unique()
```

Out[229…
```
array(['Dior', 'Unbranded', 'Giorgio Armani', 'Lattafa',
       'Maison Alhambra', 'Gucci', 'Ralph Lauren', 'Dolce & Gabbana',
       'Secertmu', 'Versace', 'Paco Rabanne', 'Grandeur', 'Armaf',
       'Carolina Herrera', 'Clinique', 'Dumont', 'Afnan', 'Azzaro',
       "Penhaligon'S", 'Bharara', 'Valentino', 'Guy Laroche', 'Montblanc',
       'Rasasi', 'Calvin Klein', 'Uomo', 'Givenchy', 'John Varvatos',
       'Nautica', 'Kenneth Cole', 'Tommy Hilfiger', '2Nd To None',
       'Yves Saint Laurent', 'Cologne', 'Roja', 'Metaherbal Labs',
       'Mirage Brands', 'Abercrombie & Fitch', 'Moschino',
       'Superz Budapest', 'Hermès', 'Diesel', 'Lacoste', 'Dossier',
       'Burberry', 'Michael Malul', 'Zara', 'Aramis',
       'Jean Paul Gaultier', 'Davidoff', 'Bvlgari', 'Parfums De Marly',
       'Salvatore Ferragamo', 'Ard Al Zaafaran', 'Karl Lagerfeld',
       'J. Del Pozo', 'Sean John', 'Ysl', 'Jaguar', 'Ebc',
       'Bath & Body Works', 'Issey Miyake', 'King', 'Prada', 'Hugo Boss',
       'Mont Blanc', 'Tommy Bahama', 'Paul Sebastian', 'Halloween',
       'Boucheron', 'Thierry Mugler', 'Jo Malone', 'Khadlaj',
       'Louis Vuitton', 'Creed', 'MFK', 'Hermes', 'Fragrance', 'Chanel',
       'Lalique', 'Liz Claiborne', 'Al Hambra', 'Joop', 'Ted Lapidus',
       'Axe', 'Lomani', 'King Of Kings', 'Rue21', 'Tom Ford',
       'Al Wataniah', 'Macarena', 'Bentley', 'Coach', 'Coty', 'Lanvin',
       'Claude Marsal', 'Nikos', 'Lucianno', 'Viktor & Rolf', 'Rochas',
       'Reyane Tradition', 'Myrurgia', 'Jovan', 'Coty Inc.',
       'Mont Blanc Legend', 'Ulric De Varens', 'Fragrance World',
       'Allsaints', 'Avon', 'Yves De Sistelle', 'Limited Edition',
       'Kenneth Cole Reaction', 'Bond No. 9', 'Acqua Di Parma',
       'Guerlain Paris', 'Al Haramain', 'Narciso Rodriguez', 'Topshelf',
       'Brut', 'Ed Hardy', 'Hollister', 'Fm', 'Paris Hilton',
       'Heaven Scents', 'Kenzo', 'Mercedes Benz', 'Franck Olivier',
       'Missoni', 'Halston', 'Baron', 'Adidas', 'Guerlain',
       'Acqua Di Gio', 'Old Spice', 'Clive Christian', 'Victor & Rolf',
       'Roberto Cavalli', 'Emanuelle Ungaro', 'Curve', 'Estée Lauder',
       'Pierre Cardin', 'Rawchemistry', 'Sterling', 'Territoire',
       'Jimmy Choo', 'Lapidus', 'Mary Kay', 'L’Occitane', 'Have A Scent',
       'Fragrance Couture', 'Mancera', 'Llure Sx', 'Jacques Bogart', 'Fc',
       'Pheromones', 'Cartier', 'Assorted', 'Hinode', 'Michael Jordan',
       'Designer Series', 'Alexandria Fragrances', 'Maison Margiela',
       'El Ganso', 'Victor Manuelle', 'English Laundry', 'Luxury',
       'Jil Sander', 'Hybrid & Company', 'Perry Ellis', 'Houbigant',
       'Tiffany', 'Emper', 'Montale', 'Fragrance One', 'Elixir',
       'Xerjoff', 'United Scents', 'Aladdin', 'Hanae Mori',
       'Phillips-Van Heusen', 'Elizabeth Taylor', 'Al Rehab', 'Bod Man',
       'Amouage', 'Dana', 'Dvyne Fragrances', 'Arabian Oud',
       'David Beckham', 'Falic Fashion Group', 'Michel Germain',
       'The Baron', 'Nina Ricci', 'Vince Camuto', 'Juicy Couture', 'Cuba',
       'Mandarin Duck', 'Guess', 'Estee Lauder', 'Philosophy',
       'Marc Jacobs', 'Qrc', 'Crepe Erase', 'Ex Nihilo',
       'Juliette Has A Gun', 'Ariana Grande', 'Sol De Janeiro', 'Kilian',
       'Donna Karan', 'Elizabeth Arden', 'Urban Outfitters',
       'Alt Fragrances', "Victoria'S Secret", 'Kim Kardashian', 'Gap',
       'Cacharel', 'Jessica Mcclintock', 'Alfred Sung',
       'Gloria Vanderbilt', 'Parfums', 'Ouai', 'Lancome',
       'Ellis Brooklyn', 'Byredo', 'Huda Beauty Kayali', 'Evyan',
       'Lake&Skye', 'Frederic Malle', 'Marilyn Miglin', 'Aquolina',
       'Katy Perry', 'Atelier Cologne', 'Christian Audigier', 'Vera Wang',
       'Clean', 'Britney Spears', 'Escada', 'Michael Kors', 'Le Labo',
       'Glossier', 'Chloe', 'Natalie', 'Chaka By Chaka Khan',
       'Huda Beauty', 'Billie Eilish', 'Lucky', 'Tf', 'Oscar De La Renta',
       'Elizabeth And James Nirvana', 'Flower', 'Parfums Gres',
       'Vicki Tiel', 'Fragonard', 'Kate Spade New York', 'Spiritual Sky',
       'True Religion', 'Taylor Swift', 'M·A·C', 'Robert Piguet',
       'Gloss Moderne', 'Justin Bieber', 'Revlon', 'Sarah Jessica Parker',
       'Jennifer Lopez', 'Kate Spade', 'J Lo', 'Yardley London',
       'Pure Romance', 'Tous', 'Parfum', 'Adam Levine', 'Jean Couturier',
       'Chopard', 'Asst', 'Al-Rehab', 'Tiziana Terenzi', 'Tory Burch',
       'Miraclelayer', 'One Direction', 'Prince Matchabelli',
       'Giardini Di Toscana', 'Jessica Simpson', 'Love Luxe Beauty',
       'Tru Fragrance', 'Sabrina Carpenter', 'Nest', 'Paloma Picasso',
       'Aeropostale', 'Phlur', 'Ainash Perfums', 'Diptyque', 'T.O.V.A.',
       'Benetton', 'Tiffany & Co.', 'Fred Hayman', 'Miim Miic', 'Aerin',
       'Pleasing', 'Pink', 'Ulta', 'Rihanna', 'Vilhelm Parfumerie',
       'Preferred Fragrance', 'Sisley', 'Anna Sui',
       'Fuller Armand Dupree', 'Atlantic', "L'Artisan", 'Clubman Pinaud',
       'Banana Republic', 'Lolita Lempicka', 'Baby Phat', 'Mem',
       'Gwen Stefani', 'Chloé', 'Jeanne Arthes', 'Daddy Yankee',
       'Classic Erotica', 'Pacifica', 'Snif', 'The Body Shop',
       'Five Star Fragrance', 'Pink Sugar', 'Elizabeth & James',
       'Elie Saab', 'Armand Dupree Fuller', 'Dkny',
       'United Colors Of Benetton'], dtype=object)
```

In [230…
```
perfumes['availableText'].unique()
```

Out[230…
```
array(['More than 10 available / 116 sold', '8 available / 48 sold',
       'More than 10 available / 27 sold', ..., '33 available / 58 sold',
       '3 available / 117 sold', '4 available / 51 sold'], dtype=object)
```

In [231…
```
# Fill in null values in the available column where we can interpolate the
value based on text
perfumes['available'] =
np.where(perfumes['availableText'].str.lower().str.contains('last\s*one|limit
quantity available'), 1, perfumes['available'])
perfumes['available'] =
np.where(perfumes['availableText'].str.lower().str.contains('out of
stock'), 0, perfumes['available'])

# Fill remaining values with standard placeholder values
perfumes['sold'].fillna(-1, inplace=True)
perfumes['lastUpdated'].fillna('Unknown', inplace=True)

# Create a column that indicates whether the number available is exact or
approximate
perfumes['availabilityExact'] =
np.where(perfumes['availableText'].str.lower().str.contains('^\d+\savailable|
one|out of stock'), 1, 0)
```

In [232…
```
perfumes[['available', 'availableText', 'availabilityExact']].head(15)
```

Out[232…

	available	availableText	availabilityExact
0	10.0	More than 10 available / 116 sold	0
1	8.0	8 available / 48 sold	1
2	10.0	More than 10 available / 27 sold	0
3	2.0	2 available / 159 sold	1
4	1.0	Limited quantity available / 156 sold	0
5	10.0	More than 10 available / 79 sold	0
6	9.0	9 available / 39 sold	1
7	1.0	Last One / 6 sold	1
8	10.0	10 available / 17 sold	1
9	8.0	8 available / 68 sold	1
10	10.0	More than 10 available / 615 sold	0
11	7.0	7 available / 458 sold	1
12	10.0	More than 10 available / 889 sold	0
13	9.0	9 available / 63 sold	1
14	7.0	7 available / 136 sold	1

In [233…

```
# Drop unnecessary columns and rename some to be more descriptive
perfumes_clean = perfumes.drop(columns=['type', 'priceWithCurrency',
'itemLocation', 'availableText', 'title'])
perfumes_clean.rename(columns={'available':'minimumAvailable',
'type_clean':'perfumeType'}, inplace=True)

# Reorder the columns
perfumes_clean = perfumes_clean[['brand',
                                'perfumeType',
                                'price',
                                'minimumAvailable',
                                'availabilityExact',
                                'sold',
                                'lastUpdated',
                                'forMen',
                                'forWomen',
                                'unisex',
                                'shipsFromUSA',
                                'shipsFromHK',
                                'shipsFromChina',
                                'shipsFromIndia',
                                'shipsFromPakistan',
                                'shipsFromCanada',
                                'shipsFromTaiwan',
                                'shipsFromBrazil',
                                'shipsFromJapan',
                                'shipsFromBulgaria']]

perfumes_clean.head()
```

Out[233…

	brand	perfumeType	price	minimumAvailable	availabilityExact	sold	lastUpdated	forMen	fo
0	Dior	Eau De Parfum	84.99	10.0	0	116.0	May 24, 2024 10:03:04 PDT	1	
1	Unbranded	Eau De Parfum	109.99	8.0	1	48.0	May 23, 2024 23:07:49 PDT	1	
2	Unbranded	Eau De Toilette	100.00	10.0	0	27.0	May 22, 2024 21:55:43 PDT	1	
3	Giorgio Armani	Eau De Toilette	44.99	2.0	1	159.0	May 24, 2024 03:30:43 PDT	1	
4	Lattafa	Fragrance	16.91	1.0	0	156.0	May 24, 2024 07:56:25 PDT	1	

In [234…

```
# Fill remaining nulls with placeholder value
perfumes_clean['minimumAvailable'].fillna(-1, inplace=True)
perfumes_clean.isna().sum()
```

Out[234…

```
brand                0
perfumeType          0
price                0
minimumAvailable     0
availabilityExact    0
sold                 0
lastUpdated          0
forMen               0
forWomen             0
unisex               0
shipsFromUSA         0
shipsFromHK          0
shipsFromChina       0
shipsFromIndia       0
shipsFromPakistan    0
shipsFromCanada      0
shipsFromTaiwan      0
shipsFromBrazil      0
```

```
shipsFromJapan       0
shipsFromBulgaria    0
dtype: int64
```

Exploratory Data Analysis

Now that the dataset is clean, we can dive into our analysis.

In [235…

```
perfumes_clean.head()
```

Out[235…

	brand	perfumeType	price	minimumAvailable	availabilityExact	sold	lastUpdated	forMen	fo
0	Dior	Eau De Parfum	84.99	10.0	0	116.0	May 24, 2024 10:03:04 PDT	1	
1	Unbranded	Eau De Parfum	109.99	8.0	1	48.0	May 23, 2024 23:07:49 PDT	1	
2	Unbranded	Eau De Toilette	100.00	10.0	0	27.0	May 22, 2024 21:55:43 PDT	1	
3	Giorgio Armani	Eau De Toilette	44.99	2.0	1	159.0	May 24, 2024 03:30:43 PDT	1	
4	Lattafa	Fragrance	16.91	1.0	0	156.0	May 24, 2024 07:56:25 PDT	1	

In [236…

```
perfumes_clean.describe()
```

Out[236…

	price	minimumAvailable	availabilityExact	sold	forMen	forWomen	u
count	1994.000000	1994.000000	1994.000000	1994.000000	1994.000000	1994.000000	1994.00
mean	43.142618	18.371113	0.639920	598.697593	0.495486	0.495988	0.01
std	32.585874	53.699996	0.480144	2149.370822	0.500105	0.500109	0.11
min	1.990000	-1.000000	0.000000	-1.000000	0.000000	0.000000	0.00
25%	21.957500	3.000000	0.000000	14.000000	0.000000	0.000000	0.00
50%	34.040000	9.000000	1.000000	49.000000	0.000000	0.000000	0.00
75%	53.877500	10.000000	1.000000	278.000000	1.000000	1.000000	0.00
max	299.990000	842.000000	1.000000	40130.000000	1.000000	1.000000	1.00

In [237…

```
perfumes_clean.describe(include="O")
```

Out[237…

	brand	perfumeType	lastUpdated
count	1994	1994	1994
unique	327	12	1826
top	Unbranded	Eau De Parfum	Unknown
freq	132	929	126

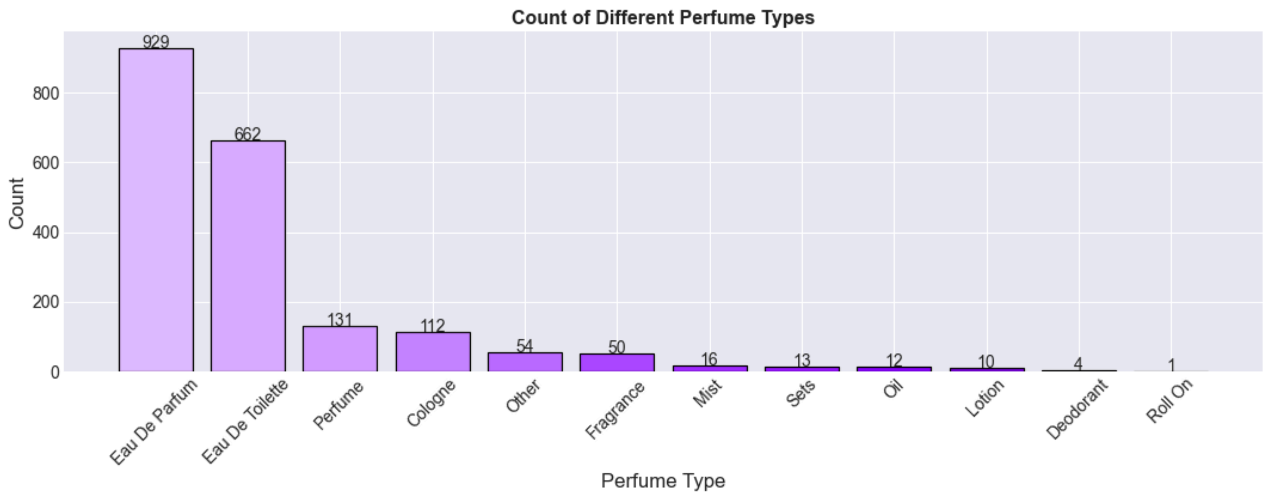
While many of these columns are considered "numerical", they actually represent categories. Most of this dataset is categorical.

Questions

What is the count of different perfume types?

In [238…

```
# What is the distribution of perfume types?
fig, ax = plt.subplots(figsize=(15, 6))
plt.bar(x=perfumes_clean['perfumeType'].value_counts().index,
height=perfumes_clean['perfumeType'].value_counts(),
color=purple_colors[::-1], edgecolor='black', linewidth=1.2)
ax.tick_params(axis='x', rotation=45)
ax.set_title('Count of Different Perfume Types')
ax.set_xlabel('Perfume Type')
ax.set_ylabel('Count')
for i in perfumes_clean['perfumeType'].value_counts().index:
    ax.annotate(f'{perfumes_clean["perfumeType"].value_counts()[i]}', xy=
(i, perfumes_clean["perfumeType"].value_counts()[i] + (.015 * ax.get_ylim()
[1])), va='center', ha='center')
plt.tight_layout();
```



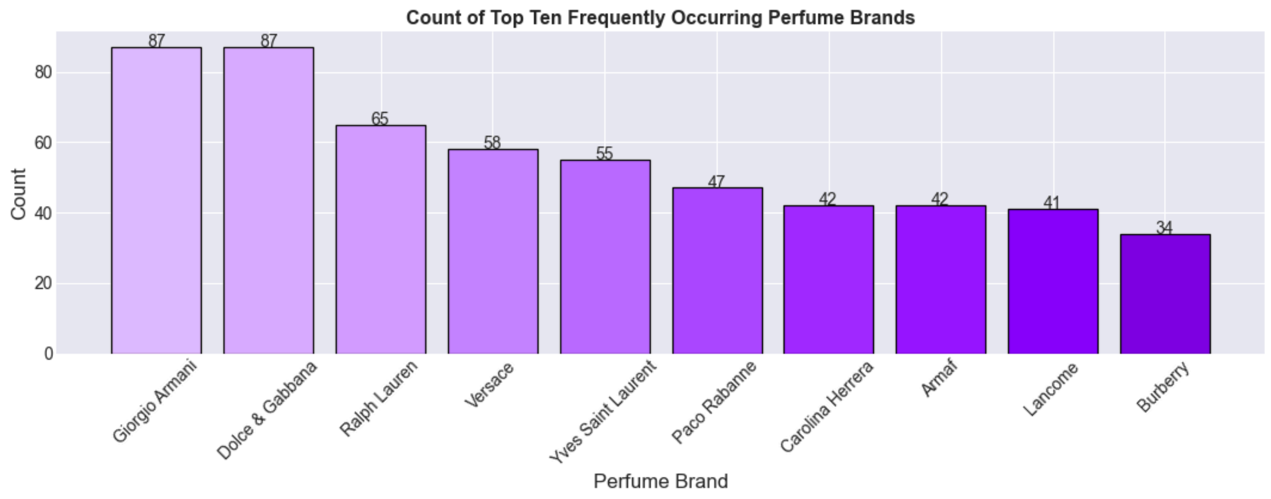
Two most frequent types of perfumes that show up on this ebay sample are Eau de Parfum and Eau de Toilette; most others are sparse.

What are the top ten most common brands?

In [239…

```python
# What is the distribution of perfume types?
data = perfumes_clean.loc[perfumes_clean['brand'] != 'Unbranded', 
'brand'].value_counts()

fig, ax = plt.subplots(figsize=(15, 6))
plt.bar(x=data.index[:10], height=data[:10], color=purple_colors[::-1], 
edgecolor='black', linewidth=1.2)
ax.tick_params(axis='x', rotation=45)
ax.set_title('Count of Top Ten Frequently Occurring Perfume Brands')
ax.set_xlabel('Perfume Brand')
ax.set_ylabel('Count')
for i in data.index:
    ax.annotate(f'{data[i]}', xy=(i, data[i] + (.015 * ax.get_ylim()[1])), 
va='center', ha='center')
plt.tight_layout();
```



The top 10 most common brands of perfumes listed in this sample of ebay are as illustrated above. Many are big name brands, with the top three most listed brands being Giorgio Armani, Dolce & Gabbana, and Ralph Lauren.

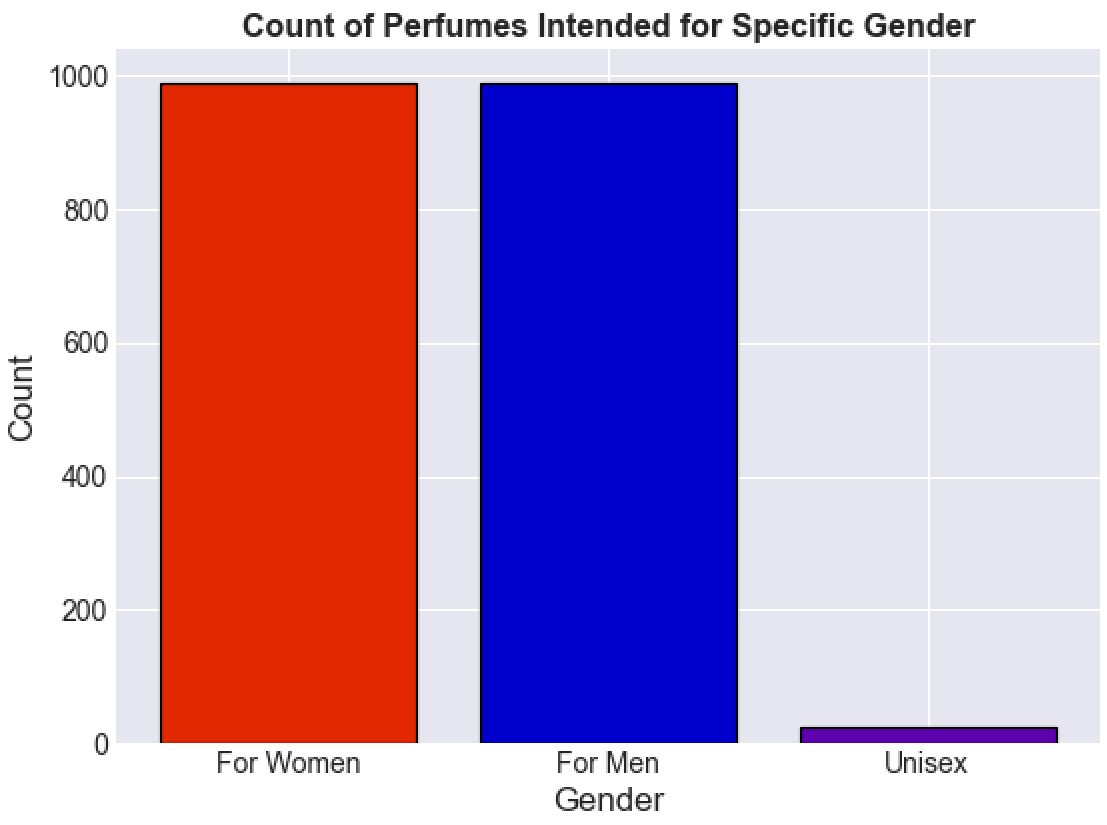
What is the distribution of perfumes intended for specific genders?

In [240…

```python
# What is the distribution of perfume types?
fig, ax = plt.subplots(figsize=(8, 6))

count_of_women = perfumes_clean['forWomen'].sum()
count_of_men = perfumes_clean['forMen'].sum()
count_of_unisex = perfumes_clean['unisex'].sum()
```

```
ax.bar(x=[1, 2, 3], height=[count_of_women, count_of_men, count_of_unisex],
color=[red_colors[5], blue_colors[5], purple_colors[5]], edgecolor='black',
linewidth=1.2)
ax.set_xticks([1, 2, 3])
ax.set_xticklabels(['For Women', 'For Men', 'Unisex'])
ax.set_title('Count of Perfumes Intended for Specific Gender')
ax.set_xlabel('Gender')
ax.set_ylabel('Count')
plt.tight_layout();
```



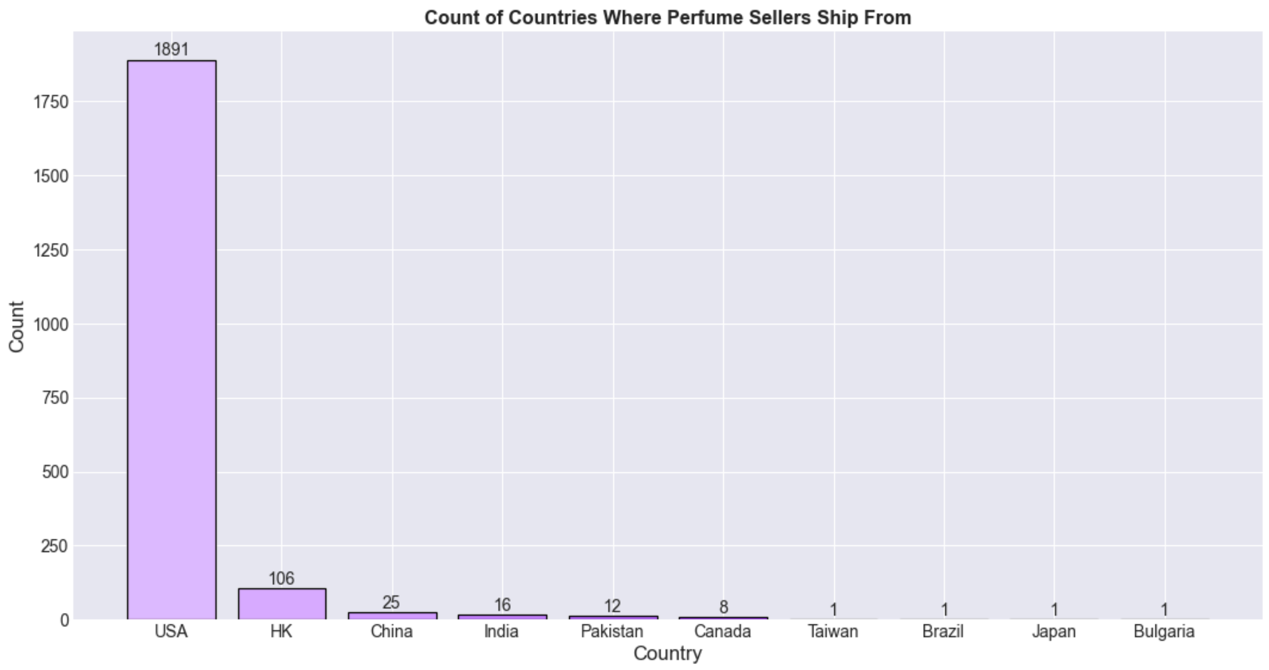
This is split very evenly down the middle, along with just a few perfumes intended for both genders.

Which locations do perfumes ship from most?

In [241…

```
fig, ax = plt.subplots(figsize=(15, 8))
locations_count = perfumes_clean.loc[:,
'shipsFromUSA':].sum().sort_values(ascending=False)
plt.bar(x=locations_count.index, height=locations_count,
color=purple_colors[::-1], edgecolor='black', linewidth=1.2)

for i in locations_count.index:
    ax.annotate(f'{locations_count[i]}', xy=(i, locations_count[i] + 30),
va='center', ha='center')
ax.set_title('Count of Countries Where Perfume Sellers Ship From')
ax.set_xlabel('Country')
ax.set_ylabel('Count')
ax.set_xticks(np.arange(len(location_categories)),
labels=location_categories)
plt.tight_layout();
```



Practically all of the observations ship from the USA, and a considerably fewer number ship from Hong Kong.

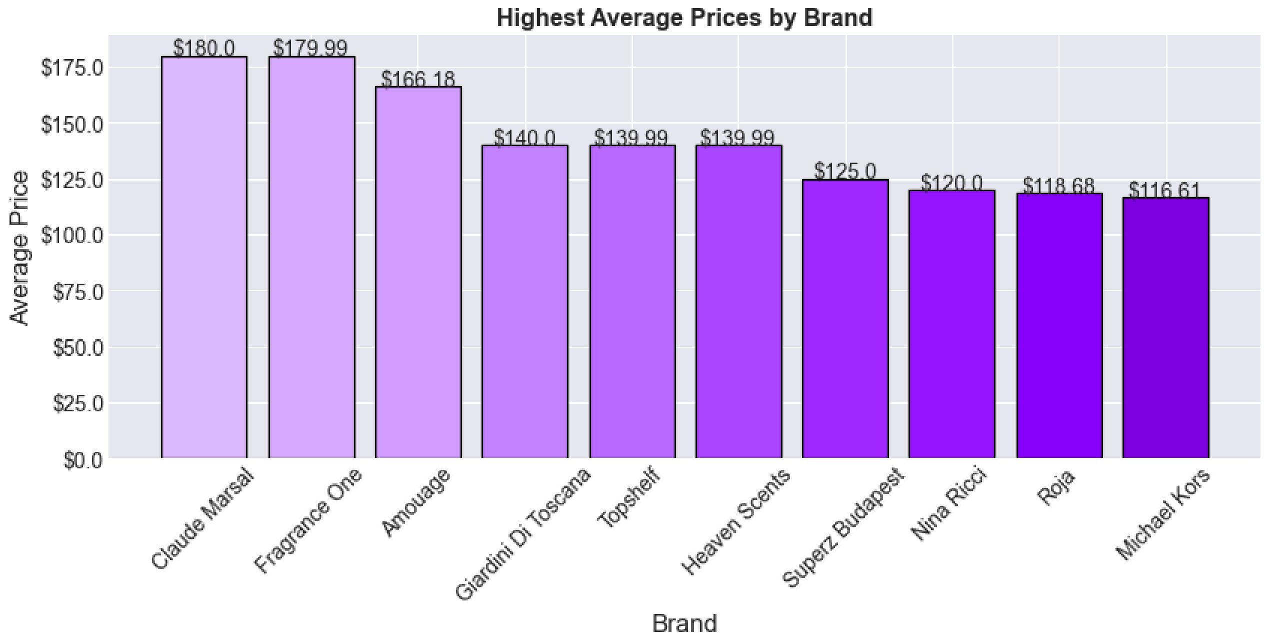
Which brands have the highest average price? Which types of perfumes have the highest average price?

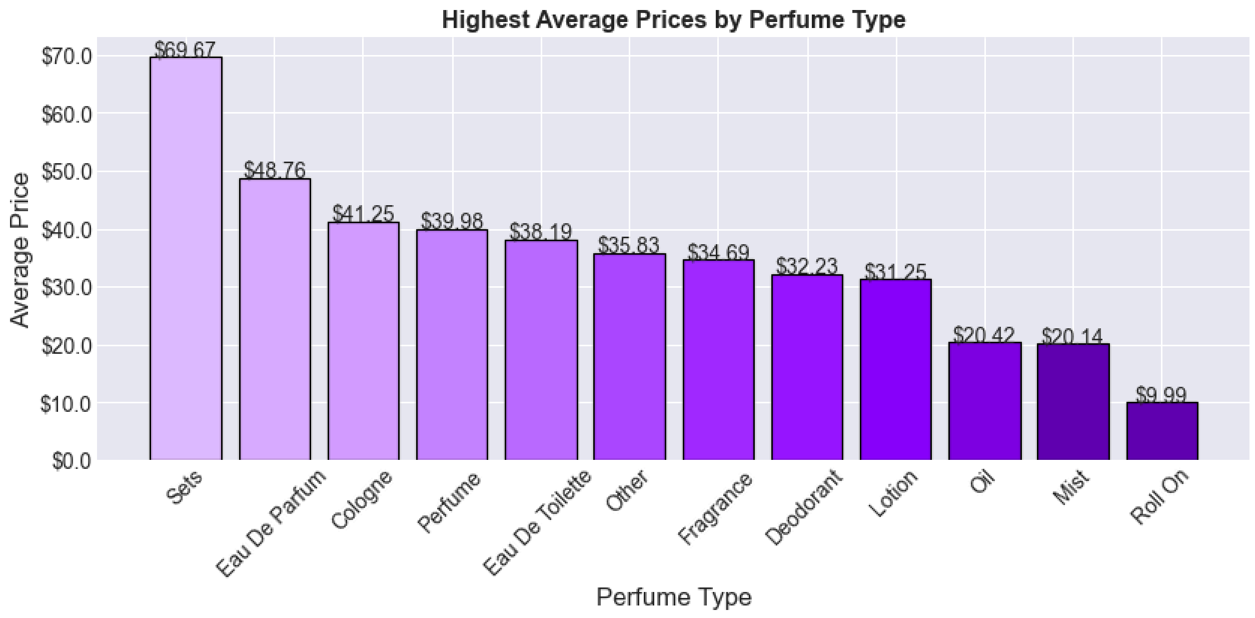
In [242…

```
def find_aggregate_columns(df, gb_column, agg_func, agg_col):
    return df.groupby(gb_column)
[agg_col].agg(agg_func).sort_values(ascending=False)

fig, ax = plt.subplots(2, 1, figsize=(12, 12))

n = 0
for col, label in [('brand', 'Brand'), ('perfumeType', 'Perfume Type')]:
    if len(find_aggregate_columns(perfumes_clean, col, 'mean', 'price')) >
15:
        plot_func = find_aggregate_columns(perfumes_clean, col, 'mean',
'price')[:10]
    else:
        plot_func = find_aggregate_columns(perfumes_clean, col, 'mean', 
'price')
    ax[n].bar(x=plot_func.index, height=plot_func, 
color=purple_colors[::-1], edgecolor='black', linewidth=1.2)
    ax[n].set_title(f'Highest Average Prices by {label}')
    ax[n].set_xlabel(f'{label}')
    ax[n].set_ylabel(f'Average Price')
    ax[n].tick_params(axis='x', rotation=45)
    for i in plot_func.index:
        ax[n].annotate(f'${plot_func[i].round(2)}', xy=(i, plot_func[i] + 
(.015 * ax[n].get_ylim()[1])), va='center', ha='center')
        ax[n].yaxis.set_major_formatter(FuncFormatter(lambda y, _: 
f'${y:}'))
    n += 1
plt.tight_layout();
```





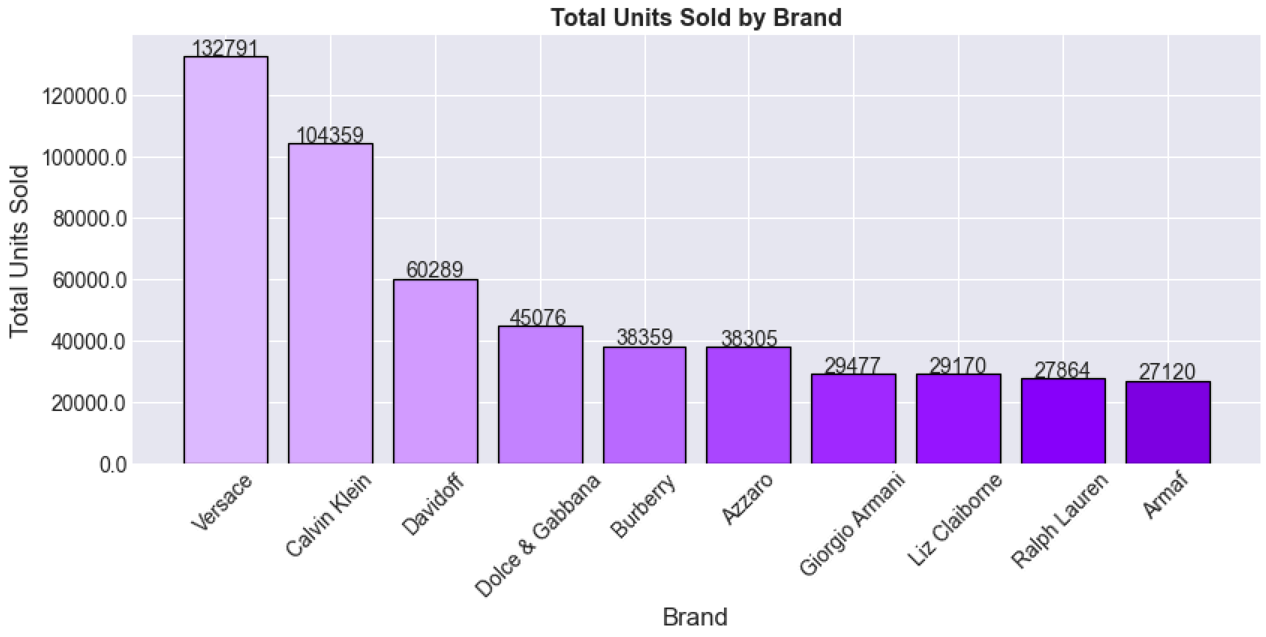
It appears that the top 10 most expensive brands on average are fairly uncommon names. None of the brands in the plot of the different brand counts made it into the top 10.

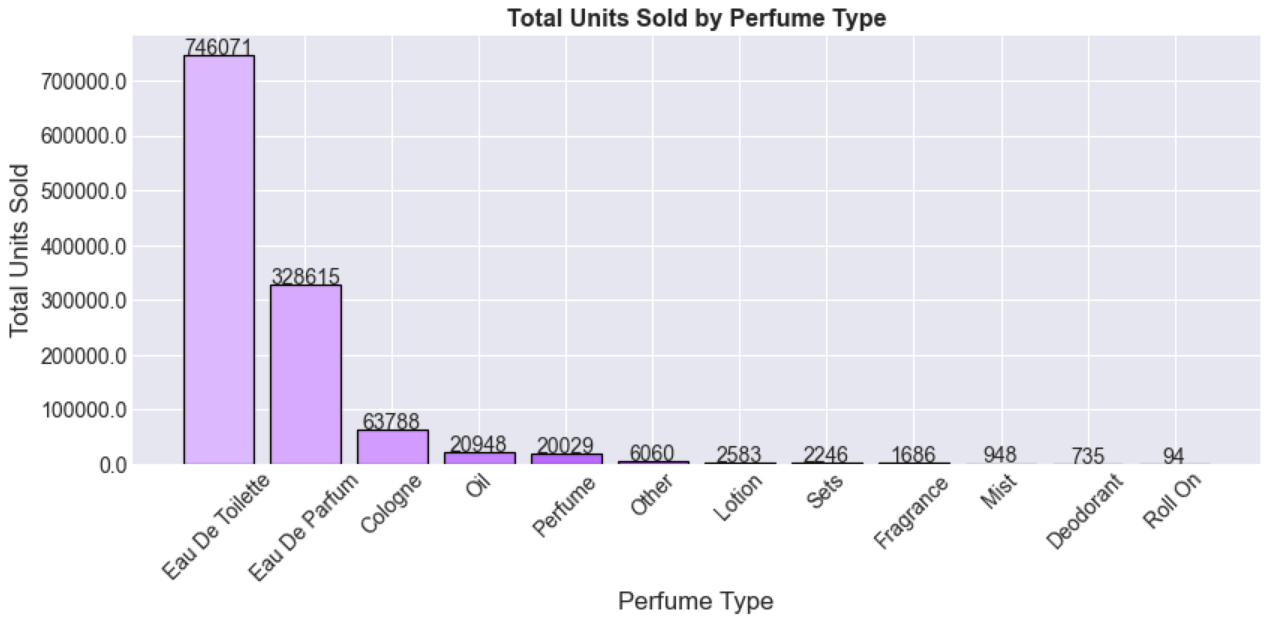
For perfume Type, while sets of perfumes made up only 13 observations in the entire dataset, it is the most expensive type of perfume that consumers buy on average. This is followed by Eau de Parfum and cologne.

What are the highest selling brands and perfume types?

In [243…
```
fig, ax = plt.subplots(2, 1, figsize=(12, 12))


n = 0
for col, label in [('brand', 'Brand'), ('perfumeType', 'Perfume Type')]:
    if len(find_aggregate_columns(perfumes_clean, col, 'sum', 'sold')) >
15:
        plot_func = find_aggregate_columns(perfumes_clean, col, 'sum',
'sold')[:10]
    else:
        plot_func = find_aggregate_columns(perfumes_clean, col, 'sum',
'sold')
    ax[n].bar(x=plot_func.index, height=plot_func,
color=purple_colors[::-1], edgecolor='black', linewidth=1.2)
    ax[n].set_title(f'Total Units Sold by {label}')
    ax[n].set_xlabel(f'{label}')
    ax[n].set_ylabel(f'Total Units Sold')
    ax[n].tick_params(axis='x', rotation=45)
    for i in plot_func.index:
        ax[n].annotate(f'{int(plot_func[i])}', xy=(i, plot_func[i]  + (.015
* ax[n].get_ylim()[1])), va='center', ha='center')
        ax[n].yaxis.set_major_formatter(FuncFormatter(lambda y, _:
f'{y:}'))
    n += 1
plt.tight_layout();
```





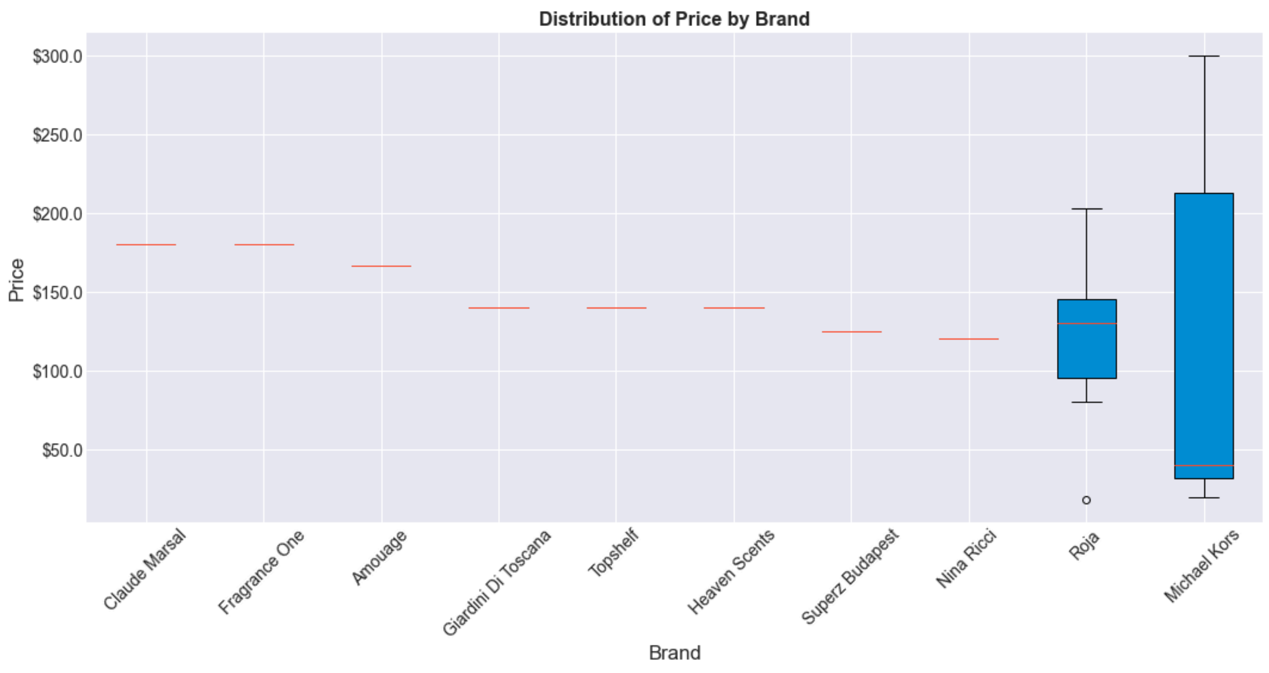
The highest selling brand is Versace and the highest selling type of perfume is Eau de Toilette. Eau De Parfum, while it has the highest average price, sells about half the units that Eau de Toilette sells.

Of the most expensive brands on average we found, what is the distribution of their prices?

In [244…

```
# What's the spread of the top brands' price?
top_brands = list(find_aggregate_columns(perfumes_clean, 'brand', 'mean',
'price').index[:10])
fig, ax = plt.subplots(figsize=(15, 8))
data = []
for brand in top_brands:
    data.append(perfumes_clean.loc[perfumes_clean['brand'].isin([brand])]
['price'])
```

```
ax.boxplot(x=data, labels=top_brands, vert=True, patch_artist=True)
ax.tick_params(axis='x', rotation=45)
ax.set_title('Distribution of Price by Brand')
ax.set_xlabel('Brand')
ax.set_ylabel('Price')
ax.yaxis.set_major_formatter(FuncFormatter(lambda y, _: f'${y:}'))
plt.tight_layout();
```



There is very little variance for the majority of these brands, probably due to there being so few observations of each. However, the last two, Roja and Michael Kors, have much more variance. Michael Kors has a huge range of values, from less than $25 up to 300$.

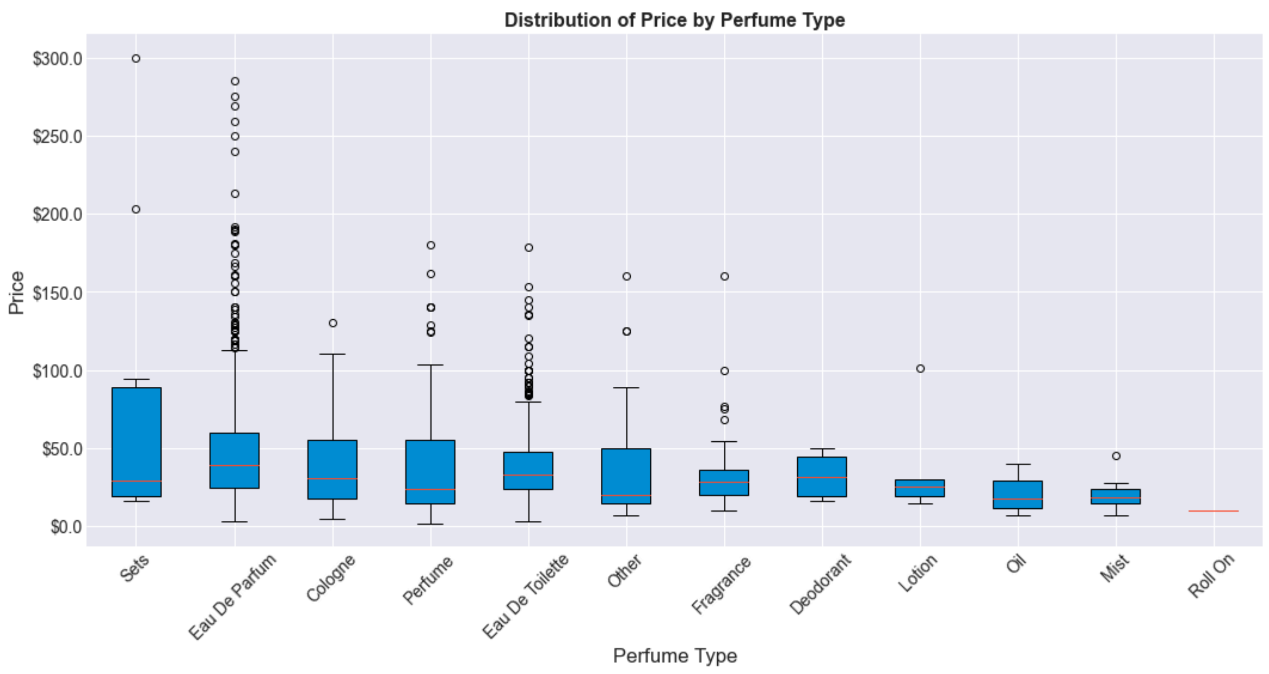
Of the different perfume types, what is the distribution of price?

In [245…

```
# What's the spread of the top types' price?
top_types = list(find_aggregate_columns(perfumes_clean, 'perfumeType',
'mean', 'price').index[:15])
fig, ax = plt.subplots(figsize=(15, 8))
data = []
for type in top_types:

data.append(perfumes_clean.loc[perfumes_clean['perfumeType'].isin([type])]
['price'])

ax.boxplot(x=data, labels=top_types, vert=True, patch_artist=True)
ax.tick_params(axis='x', rotation=45)
ax.set_title('Distribution of Price by Perfume Type')
ax.set_xlabel('Perfume Type')
ax.set_ylabel('Price')
ax.yaxis.set_major_formatter(FuncFormatter(lambda y, _: f'${y:}'))
plt.tight_layout();
```



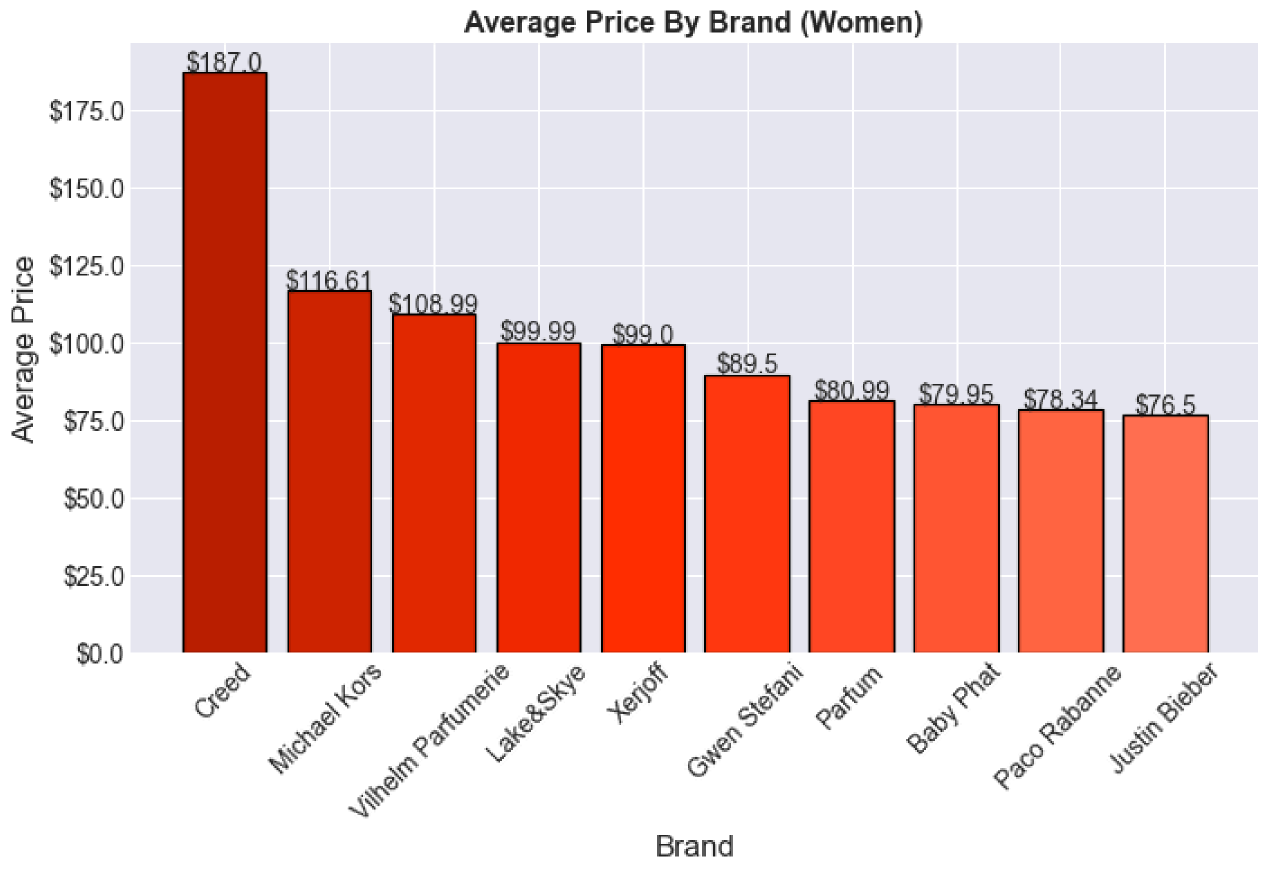
Sets have a more limited range of prices than Eau de Parfum does, but the overall range is a tad higher than Eau de Parfum. Eau de Parfum has many, many outliers as seen by the circles on top of its box.

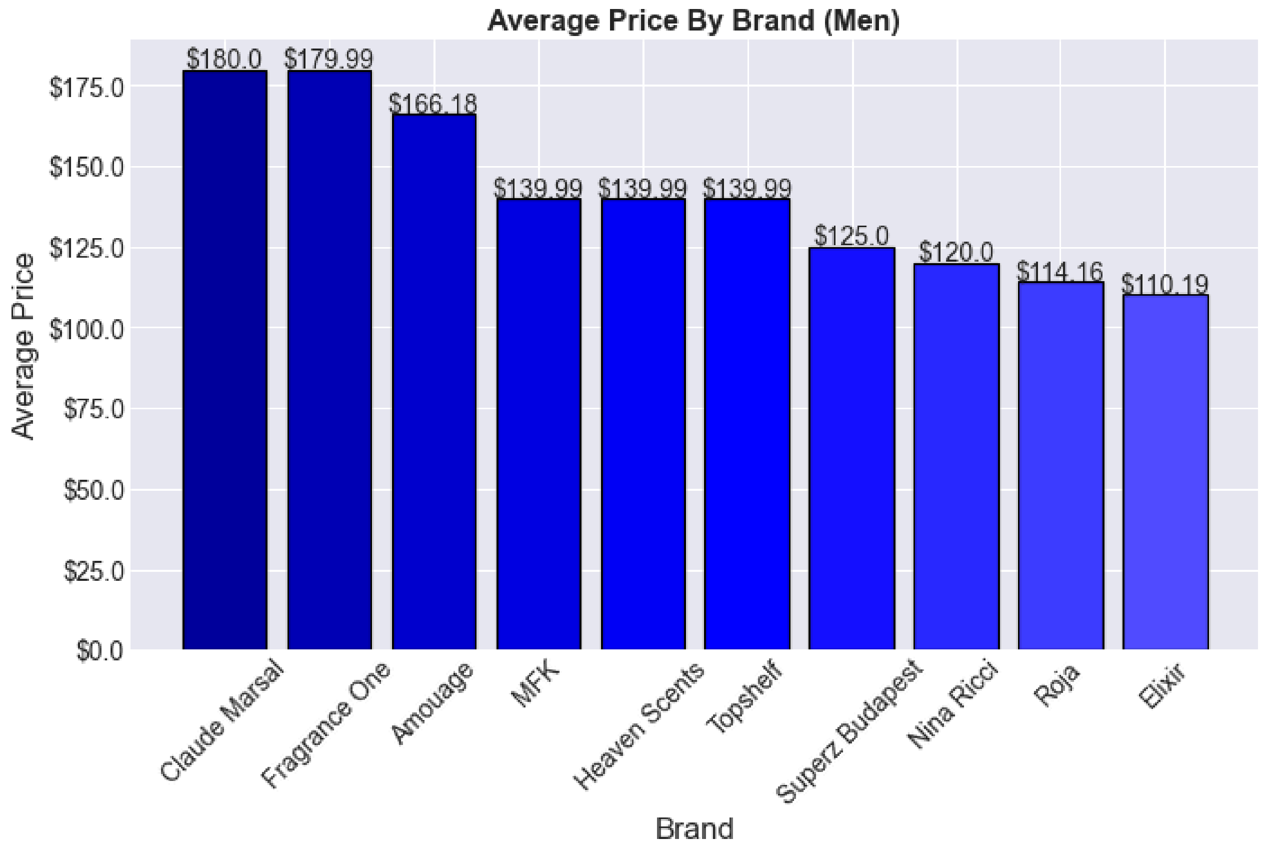
What about the highest average price of brands and perfume types across different genders?

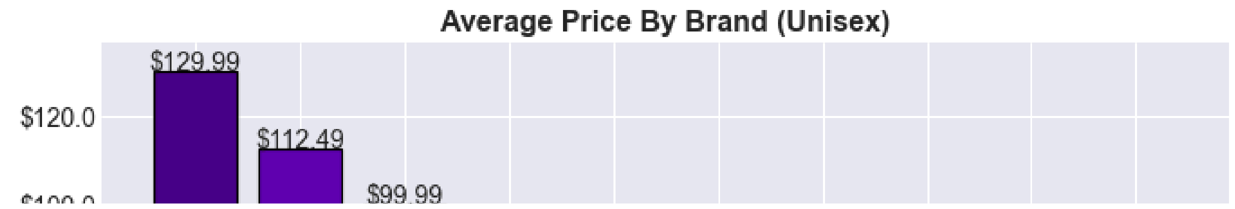
In [246…

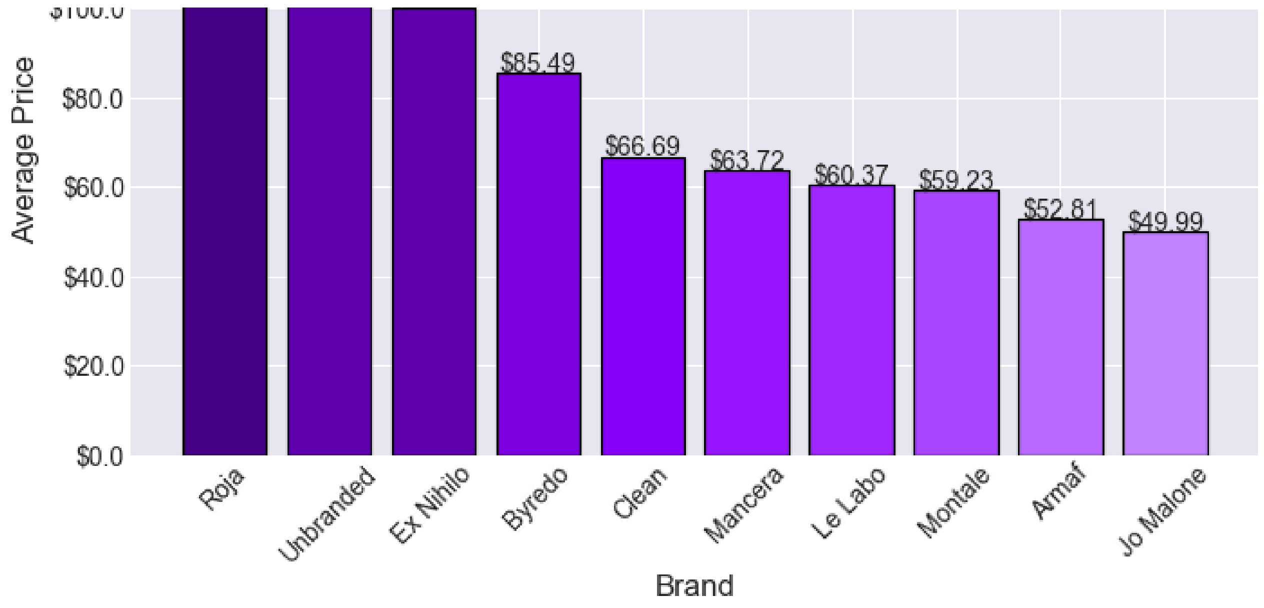
```
most_sold_brands_women =
find_aggregate_columns(perfumes_clean.loc[(perfumes_clean['sold'] >= 0) &
(perfumes_clean['forWomen'] == 1)], 'brand', 'mean', 'price')[:10]
most_sold_brands_men =
find_aggregate_columns(perfumes_clean.loc[(perfumes_clean['sold'] >= 0) &
(perfumes_clean['forMen'] == 1)], 'brand', 'mean', 'price')[:10]
most_sold_brands_unisex =
find_aggregate_columns(perfumes_clean.loc[(perfumes_clean['sold'] >= 0) &
(perfumes_clean['unisex'] == 1)], 'brand', 'mean', 'price')[:10]

fig, ax = plt.subplots(3, 1, figsize=(10, 20))
n = 0
for data, label, colors in [(most_sold_brands_women, 'Women',
red_colors[3:]), (most_sold_brands_men, 'Men', blue_colors[3:]),
(most_sold_brands_unisex, 'Unisex', purple_colors[3:])]:
    ax[n].bar(x=data.index, height=data, color=colors, edgecolor='black',
linewidth=1.2)
    ax[n].set_title(f'Average Price By Brand ({label})')
    ax[n].set_xlabel('Brand')
    ax[n].set_ylabel('Average Price')
    ax[n].tick_params(axis='x', rotation=45)
    for i in data.index:
        ax[n].annotate(f'${data[i].round(2)}', xy=(i, data[i] + (.015 *
ax[n].get_ylim()[1])), va='center', ha='center')
        ax[n].yaxis.set_major_formatter(FuncFormatter(lambda y, _:
f'${y:}'))
    n += 1
plt.tight_layout();
```


Average Price By Brand (Women)
Average Price
Brand
$0.0
$25.0
$50.0
$75.0
$100.0
$125.0
$150.0
$175.0
$187.0
$116.61
$108.99
$99.99
$99.0
$89.5
$80.99
$79.95
$78.34
$76.5
Creed
Michael Kors
Vilhelm Parfumerie
Lake&Skye
Xerjoff
Gwen Stefani
Parfum
Baby Phat
Paco Rabanne
Justin Bieber


Average Price By Brand (Men)
Average Price
Brand
$0.0
$25.0
$50.0
$75.0
$100.0
$125.0
$150.0
$175.0
$180.0
$179.99
$166.18
$139.99
$139.99
$139.99
$125.0
$120.0
$114.16
$110.19
Claude Marsal
Fragrance One
Amouage
MFK
Heaven Scents
Topshelf
Superz Budapest
Nina Ricci
Roja
Elixir


Average Price By Brand (Unisex)
$120.0
$129.99
$112.49
$99.99



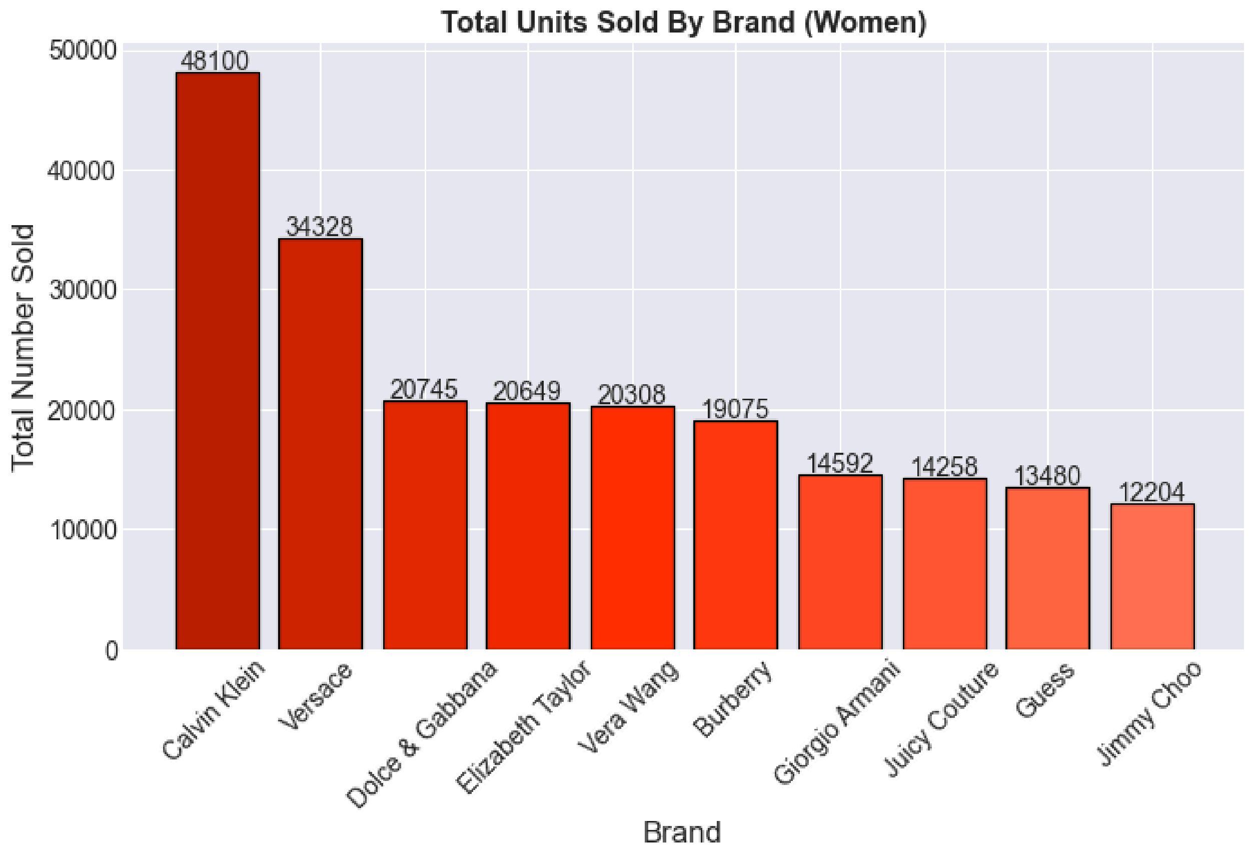
The brands with the highest average price for women, men, and unisex are Creed, Claude Marsal, and Roja respectively.

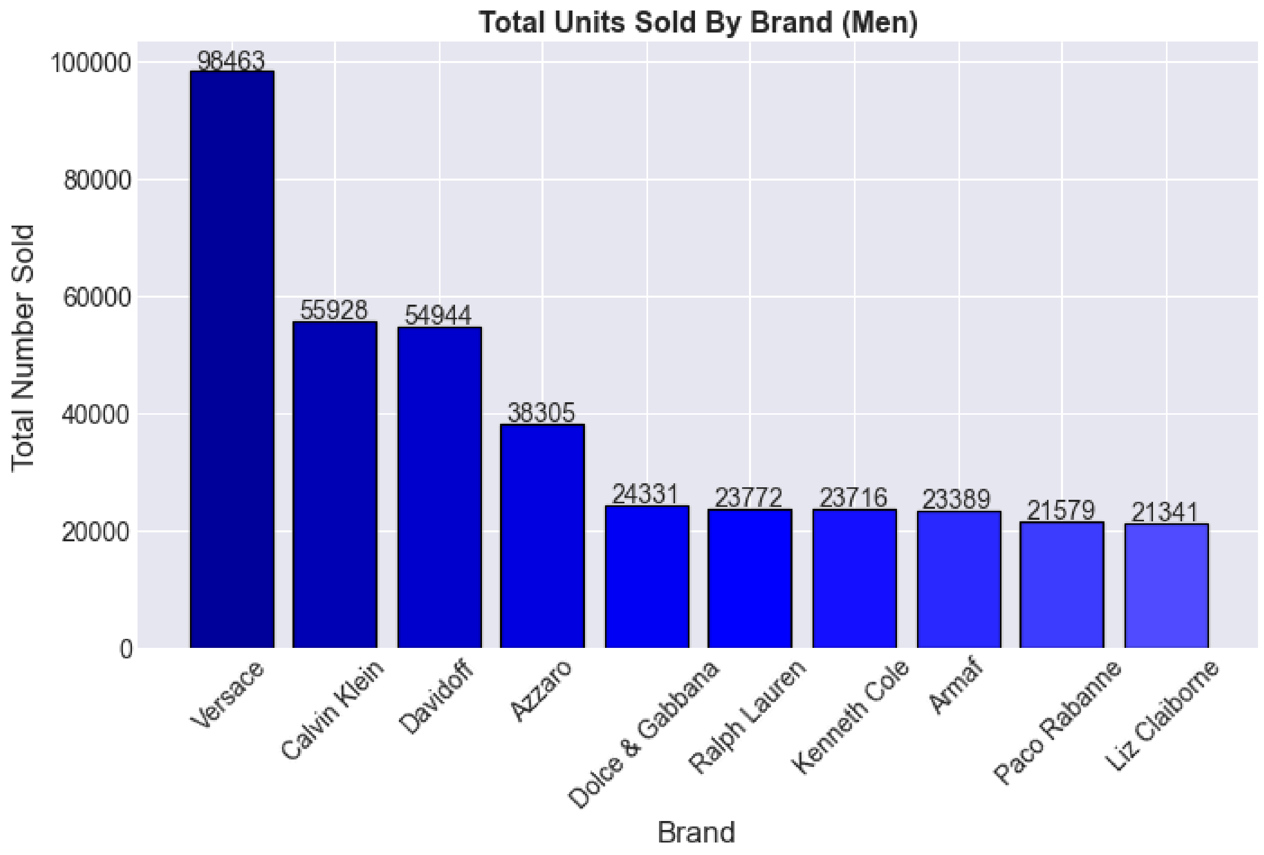
What are the top selling brands of perfumes across genders?

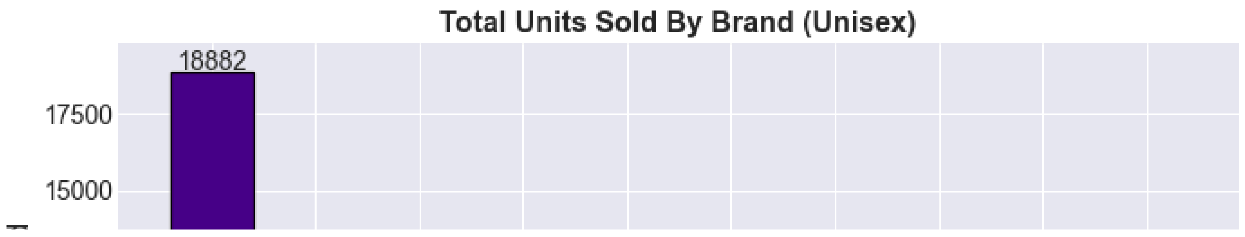
In [247…

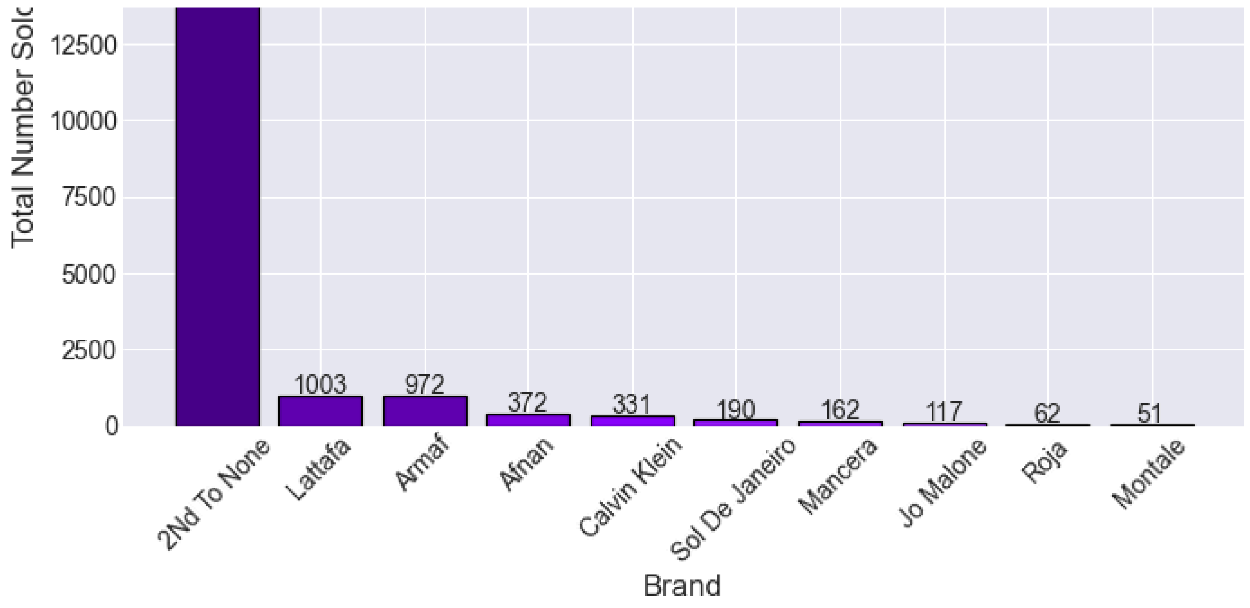
```
# What brands sell the most?
most_sold_brands_women =
find_aggregate_columns(perfumes_clean.loc[(perfumes_clean['sold'] >= 0) &
(perfumes_clean['forWomen'] == 1)], 'brand', 'sum', 'sold')[:10]
most_sold_brands_men =
find_aggregate_columns(perfumes_clean.loc[(perfumes_clean['sold'] >= 0) &
(perfumes_clean['forMen'] == 1)], 'brand', 'sum', 'sold')[:10]
most_sold_brands_unisex =
find_aggregate_columns(perfumes_clean.loc[(perfumes_clean['sold'] >= 0) &
(perfumes_clean['unisex'] == 1)], 'brand', 'sum', 'sold')[:10]

fig, ax = plt.subplots(3, 1, figsize=(10, 20))
n = 0
for data, label, colors in [(most_sold_brands_women, 'Women',
red_colors[3:]), (most_sold_brands_men, 'Men', blue_colors[3:]),
(most_sold_brands_unisex, 'Unisex', purple_colors[3:])]:
    ax[n].bar(x=data.index, height=data, color=colors, edgecolor='black',
linewidth=1.2)
    ax[n].set_title(f'Total Units Sold By Brand ({label})')
    ax[n].set_xlabel('Brand')
    ax[n].set_ylabel('Total Number Sold')
    ax[n].tick_params(axis='x', rotation=45)
    for i in data.index:
        ax[n].annotate(f'{int(data[i])}', xy=(i, data[i] + (.015 * 
ax[n].get_ylim()[1])), va='center', ha='center')
    n += 1
plt.tight_layout();
```


Total Units Sold By Brand (Women)
Total Number Sold
Brand
48100
34328
20745
20649
20308
19075
14592
14258
13480
12204
Calvin Klein
Versace
Dolce & Gabbana
Elizabeth Taylor
Vera Wang
Burberry
Giorgio Armani
Juicy Couture
Guess
Jimmy Choo


Total Units Sold By Brand (Men)
Total Number Sold
Brand
98463
55928
54944
38305
24331
23772
23716
23389
21579
21341
Versace
Calvin Klein
Davidoff
Azzaro
Dolce & Gabbana
Ralph Lauren
Kenneth Cole
Armaf
Paco Rabanne
Liz Claiborne


Total Units Sold By Brand (Unisex)
18882
17500
15000



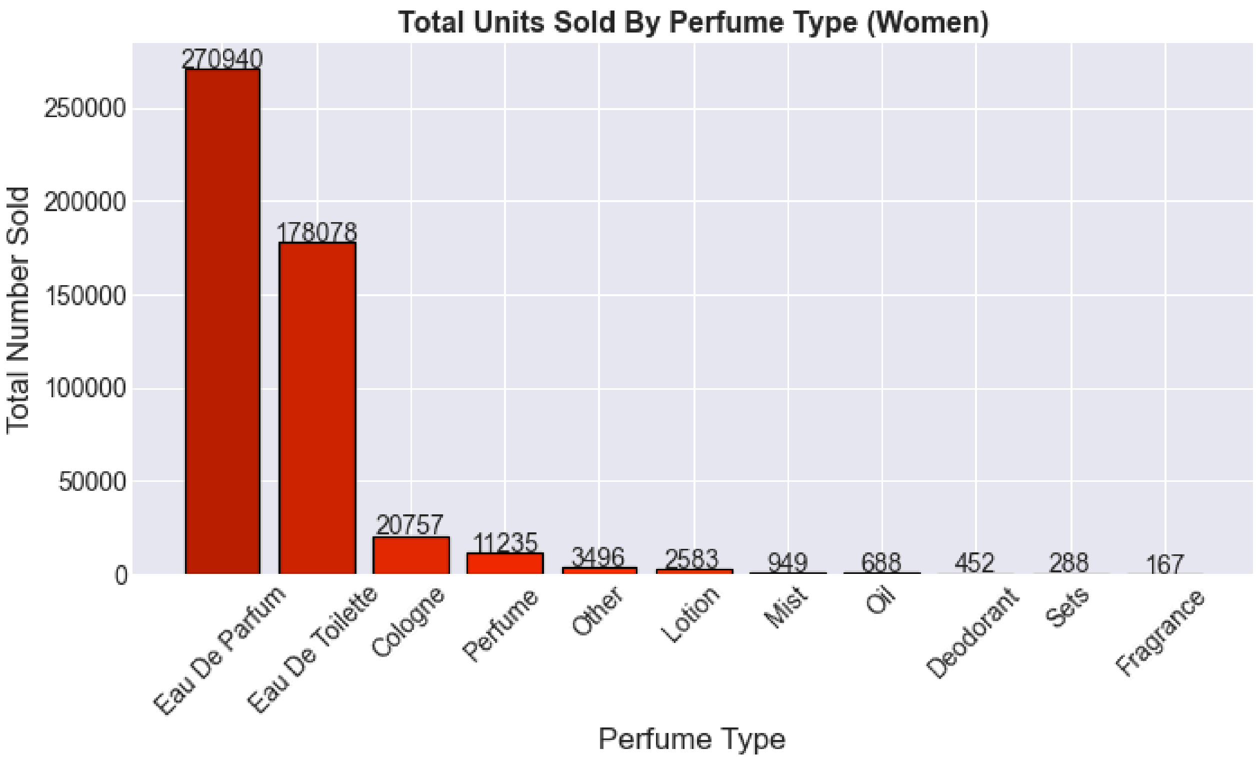
Calvin Klein is included in all three of these categories, and Versace and Calvin Klein are in first and second places for both men and women. The unisex brands are very skewed, with the majority of units sold are by the brand 2nd to None.

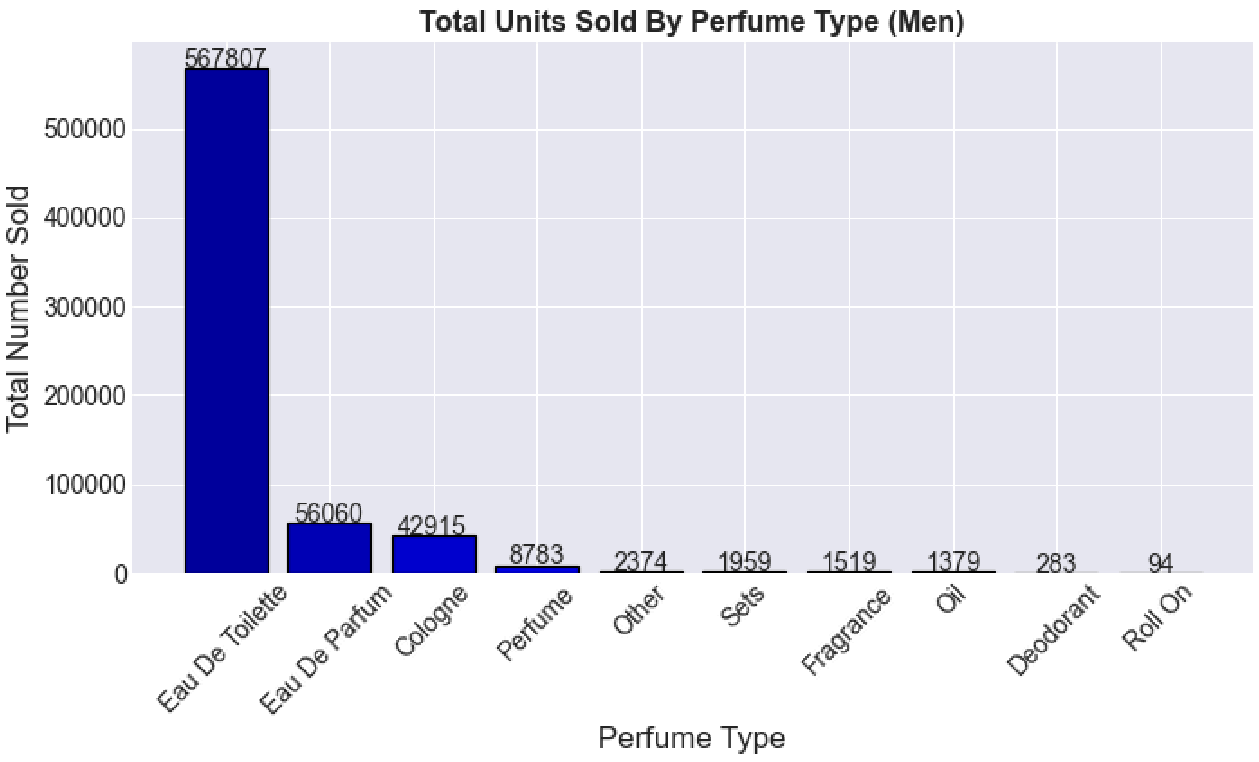
What perfume types sell the most across genders?

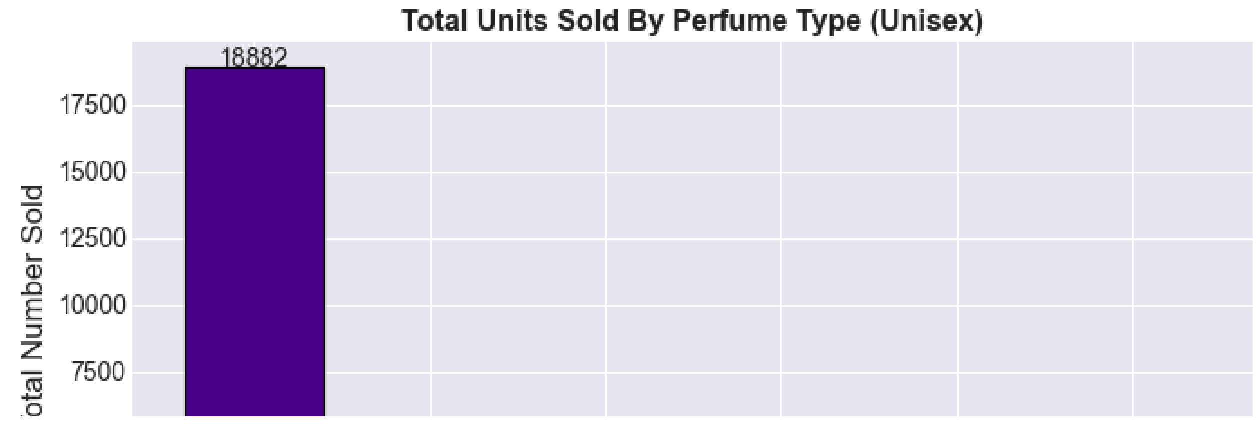
In [256…

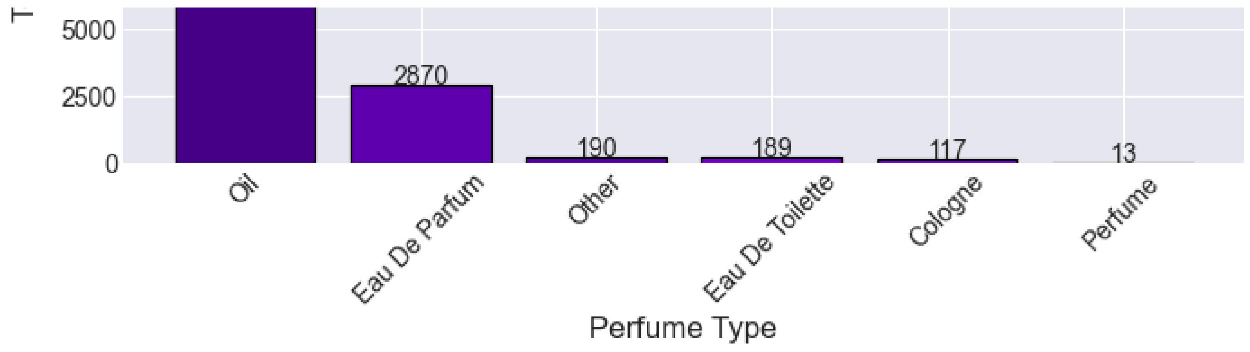
```
# What perfume types sell the most?
most_sold_brands_women =
find_aggregate_columns(perfumes_clean.loc[(perfumes_clean['sold'] >= 0) &
(perfumes_clean['forWomen'] == 1)], 'perfumeType', 'sum', 'sold')
most_sold_brands_men =
find_aggregate_columns(perfumes_clean.loc[(perfumes_clean['sold'] >= 0) &
(perfumes_clean['forMen'] == 1)], 'perfumeType', 'sum', 'sold')
most_sold_brands_unisex =
find_aggregate_columns(perfumes_clean.loc[(perfumes_clean['sold'] >= 0) &
(perfumes_clean['unisex'] == 1)], 'perfumeType', 'sum', 'sold')

fig, ax = plt.subplots(3, 1, figsize=(10, 18))
n = 0
for data, label, colors in [(most_sold_brands_women, 'Women',
red_colors[3:]), (most_sold_brands_men, 'Men', blue_colors[3:]),
(most_sold_brands_unisex, 'Unisex', purple_colors[3:])]:
    ax[n].bar(x=data.index, height=data, color=colors, edgecolor='black',
linewidth=1.2)
    ax[n].set_title(f'Total Units Sold By Perfume Type ({label})')
    ax[n].set_xlabel('Perfume Type')
    ax[n].set_ylabel('Total Number Sold')
    ax[n].tick_params(axis='x', rotation=45)
    for i in data.index:
        ax[n].annotate(f'{int(data[i])}', xy=(i, data[i] + (.015 *
ax[n].get_ylim()[1])), va='center', ha='center')
    n += 1
plt.tight_layout();
```


Total Units Sold By Perfume Type (Women)
Total Number Sold
Perfume Type
0
50000
100000
150000
200000
250000
Eau De Parfum 270940
Eau De Toilette 178078
Cologne 20757
Perfume 11235
Other 3496
Lotion 2583
Mist 949
Oil 688
Deodorant 452
Sets 288
Fragrance 167


Total Units Sold By Perfume Type (Men)
Total Number Sold
Perfume Type
0
100000
200000
300000
400000
500000
Eau De Toilette 567807
Eau De Parfum 56060
Cologne 42915
Perfume 8783
Other 2374
Sets 1959
Fragrance 1519
Oil 1379
Deodorant 283
Roll On 94


Total Units Sold By Perfume Type (Unisex)
otal Number Sold
7500
10000
12500
15000
17500
18882



For perfume type intended for women, the most popular perfume type is Eau de Parfum, while for men, the most popular type is Eau de Toilette. Oil is the most popular type of perfume for the perfumes intended for both genders.

In [249…

```
most_sold_brands_type_women =
find_aggregate_columns(perfumes_clean.loc[(perfumes_clean['sold'] >= 0) &
(perfumes_clean['forWomen'] == 1)], ['perfumeType', 'brand'], 'sum',
'sold')
most_sold_brands_type_men =
find_aggregate_columns(perfumes_clean.loc[(perfumes_clean['sold'] >= 0) &
(perfumes_clean['forMen'] == 1)], ['perfumeType', 'brand'], 'sum', 'sold')
most_sold_brands_type_unisex =
find_aggregate_columns(perfumes_clean.loc[(perfumes_clean['sold'] >= 0) &
(perfumes_clean['unisex'] == 1)], ['perfumeType', 'brand'], 'sum', 'sold')
```

Key Takeaways:

- For perfume intended towards women, the top brand and top type to sell are Calvin Klein, and Eau de Parfum.
- For perfume intended towards men, the top brand and top type to sell are Versace, and Eau de Toilette.
- For perfume intended for both genders, the top brand and top type to sell are 2nd to None, and Oil.

Side note:

2nd to None is actually a seller on ebay, not a major perfume brand. Unless you would be able to buy wholesale from them and sell their products as a third party, they will be your biggest competition.

In [255…

```
print(f"The optimal combination for mens perfume is:
{most_sold_brands_type_men.index[0]}, it sold
{(most_sold_brands_type_men[0])} units.")
print(f"The optimal combination for womens perfume is:
```

```
{most_sold_brands_type_women.index[0]}, it sold
{(most_sold_brands_type_women[0])} units.")
print(f"The optimal combination for unisex perfume is:
{most_sold_brands_type_unisex.index[0]}, it sold
{(most_sold_brands_type_unisex[0])} units.")
```

```
The optimal combination for mens perfume is: ('Eau De Toilette', 'Versace'), it sold 960
89.0 units.
The optimal combination for womens perfume is: ('Eau De Parfum', 'Calvin Klein'), it sol
d 43892.0 units.
The optimal combination for unisex perfume is: ('Oil', '2Nd To None'), it sold 18882.0 u
nits.
```