المحتويات

التمثيل المرئي للبيانات: نافذة لفهم العالم من حولنا

في عالمنا الغني بالمعلومات، تُصبح البيانات عنصراً أساسياً لاتخاذ القرارات وفهم الظواهر المختلفة. لكن البيانات، في حالتها الخام، قد تكون مربكة وصعبة الفهم. هنا يأتي دور **التمثيل المرئي للبيانات** ليُلعب دوراً هاماً في تحويل البيانات إلى معلومات سهلة الاستيعاب والتفسير.

ما هو التمثيل المرئي للبيانات؟

يتضمن التمثيل المرئي للبيانات استخدام تقنيات بصرية مثل الرسوم البيانية والجداول والخرائط لعرض البيانات بطريقة واضحة وجذابة. فهو يُساعدنا على:

- **فهم العلاقات والاتجاهات:** تُظهر لنا الرسوم البيانية والجداول كيف تترابط البيانات مع بعضها البعض وكيف تتغير بمرور الوقت.
- **اكتشاف الأنماط والقيم المتطرفة:** يمكننا بسهولة تحديد الأنماط غير العادية أو القيم المتطرفة في البيانات من خلال تمثيلها بصرياً.
- **تسهيل التواصل:** تُساعدنا الرسوم البيانية والجداول على مشاركة المعلومات مع الآخرين بشكل فعال، حتى مع أولئك الذين ليس لديهم خلفية تقنية.
- **تحفيز الإبداع:** يُمكن أن يُلهمنا التمثيل المرئي للبيانات للتفكير بشكل إبداعي وإيجاد حلول جديدة للمشكلات.
- **تسهيل اتخاذ القرارات:** تُساعدنا المعلومات المُقدمة من خلال التمثيل المرئي للبيانات على اتخاذ قرارات أفضل وأكثر ذكاءً.

أمثلة على استخدامات التمثيل المرئي للبيانات:

- **في مجال الأعمال :**تستخدم الشركات التمثيل المرئي للبيانات لتحليل مبيعاتها وأرباحها وتقييم أداء موظفيها وفهم سلوك عملائها.

- **في مجال العلوم :**يستخدم العلماء التمثيل المرئي للبيانات لعرض نتائج أبحاثهم وفهم الظواهر الطبيعية وتتبع تغيرات المناخ.
- **في مجال الطب :**يستخدم الأطباء التمثيل المرئي للبيانات لتشخيص الأمراض ومتابعة صحة المرضى واتخاذ قرارات العلاج.
- **في مجال الصحافة :**تستخدم وسائل الإعلام التمثيل المرئي للبيانات لعرض الأخبار والمعلومات بطريقة جذابة ومفهومة.
- **في مجال التعليم :**يستخدم المعلمون التمثيل المرئي للبيانات لشرح المفاهيم المعقدة للطلاب وتسهيل فهم المواد الدراسية.

يُعدّ التمثيل المرئي للبيانات أداة أساسية لفهم البيانات والتواصل بها بشكل فعال. من خلال اختيار المخطط المناسب واستخدام تقنيات التصميم الجيدة، يمكنك تحويل البيانات إلى معلومات مفيدة قابلة للتنفيذ.

في هذا البحث سنتناول عشرة مخططات مستخدمة في التمثيل المرئي للبيانات بناءاً على تحليل مجموعة بيانات مناسبة وباستخدام عدة مكتبات تدعمها لغة بايثون.

لمحة عامة عن المخططات العشرة التي تم اختيارها :

1. **الخريطة الحرارية (Heatmap):** مخطط يستخدم الألوان لتمثيل البيانات ثنائية البعد، حيث تظهر القيم الأعلى بألوان ساخنة والقيم الأدنى بألوان باردة.
2. **الخريطة الشجرية (Treemap)** :مخطط يقسم المساحة إلى مستطيلات بأحجام متناسبة مع قيم البيانات، مما يوفر نظرة شاملة على التسلسل الهرمي للبيانات.
3. **المخطط الشلالي (Waterfall)** :مخطط يوضح التغيرات التراكمية في قيمة ما على مدار فترة زمنية، مع إظهار الزيادات والنقصان بشكل واضح.
4. **مخطط الفقاعات (Bubble)** : مخطط يستخدم الدوائر لتمثيل البيانات ثلاثية الأبعاد، حيث حجم الدائرة يمثل قيمة البيانات والموضع الأفقي والرأسي يمثل البعدين الآخرين.
5. **مخطط أشعة الشمس (Sunbrust) :** مخطط دائري هرمي يستخدم لتمثيل البيانات على شكل تسلسل هرمي، مع إظهار العلاقات بين المستويات المختلفة.

6. **مخطط ساينكي (Sankey)** : مخطط يوضح تدفق البيانات بين مختلف المكونات، حيث يتم تمثيل التدفق بأشرطة عرضها متناسب مع حجم التدفق.
7. **المخطط الدائري (Pie)** : مخطط دائري يستخدم لتمثيل البيانات كنسب مئوية من إجمالي القيمة.
8. **المخطط الشريطي الأفقي (Horizontal bar)** : مخطط يستخدم أشرطة أفقية لتمثيل البيانات، مناسب لمقارنة القيم.
9. **المخطط الشريطي العمودي (Column)** : مخطط يستخدم أعمدة رأسية لتمثيل البيانات، مناسب لمقارنة القيم.
10. **المخطط الشريطي المقارن**: مخطط يستخدم أشرطة رأسية أو أفقية لتمثيل وقارن بيانات متعددة في نفس الوقت.

تم العمل في بيئة Jupyter Notebook :

Jupyter Notebook: هو تطبيق ويب مفتوح المصدر يستخدم لإنشاء وتشغيل وتبادل المستندات التفاعلية. هذه المستندات تتكون من خلايا كود قابلة للتحرير والتنفيذ، ويمكن أيضاً إدراج نصوص وصور ورسومات وغيرها.

بعض المميزات الرئيسية لـJupyter Notebook:

1 .متعدد اللغات: يدعم العديد من لغات البرمجة مثلPython ، R، Julia، وغيرها.

2 .بيئة تفاعلية: يسمح بتحرير وتنفيذ الشفرة البرمجية والحصول على النتائج مباشرة في نفس المستند.

3 .إمكانية التشارك والتعاون: يمكن مشاركة المستندات والعمل عليها بشكل تشاركي.

4 .دعم الوسائط المتعددة: يمكن إدراج رسومات وصور وفيديو داخل المستندات.

5 .إمكانية النشر والتوزيع: يمكن تصدير المستندات إلى تنسيقات مختلفة كـ HTML أو.PDF

Jupyter Notebook يُعد أداة قوية للاستكشاف والتحليل البيانات، والتعلم الآلي، والبحث العلمي، والتعليم، وغيرها من التطبيقات. إنه يوفر بيئة تطوير متكاملة وسهلة الاستخدام للمطورين والباحثين.

تمثيل مجموعة بيانات لرسم المخططات البيانية:

يوفر موقع www.kaggle .com مجموعة بيانات كبيرة من بيانات التدريب لطيف واسع من مسائل تحليل البيانات وتعلم الآلة. لذا تم اختيار مجموعة بيانات من هذا الموقع بما يتناسب مع المخططات التي تم اختيارها من مقرر التمثيل المرئي DDV601.

:Online Elctronic Sales

توفر مجموعة البيانات هذه نظرة عامة شاملة على معاملات مبيعات الالكترونيات عبر الإنترنت. يمثل كل صف معاملة واحدة تحتوي على معلومات تفصيلية مثل معرف الطلب وتاريخ تقديم الطلب وبلد المشتري ومدينته والفئات التي تم شراؤها و الفئات الفرعية منها وسعر الوحدة والكميات المباعة منها والخصومات المقدمة للمشتري والتكلفة الاجمالية لكل فئة .

المعاينة الأولية للبيانات:

يمكن معاينة ملف بصيغة csv في Excel لأخذ فكرة أولية عن البيانات:

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P
1	Order ID,Order Date,Day,Country,City,Lat,Lng,Full Name,Category,Sub Category,Item,SalesPerson ID,Quantity,Unit Price,Discount,Total Cost,Status,Gender															
2	1,01/01/2023,Mon,Syria,homs,34.7326,36.7136,Lina Alrrashid,Tablet,Apple iPad,"iPad Pro 12.9""",N498,4,999,38.3616,891.9072,False,Female															
3	2,01/01/2023,Tue,Saudi Arabia,riyadh,24.7136,46.6753,Omar Eurul,Smartphone,Samsung Galaxy,Galaxy S21 Ultra,X918,3,1199,517.968,302.148,True,Female															
4	3,01/01/2023,Tue,Saudi Arabia,riyadh,24.7743,46.7386,Iman Iismaeil,Digital Camera,Panasonic Lumix,Panasonic Lumix GH5,I036,4,1299,883.32,831.36,True,Male															
5	4,01/01/2023,Mon,United Arab Emirates,abu dhabi,24.4539,54.3773,Ahmad Rihan,Tablet,Samsung Galaxy Tab,Galaxy Tab A8,E804,6,199,33.3126,129.549,True,Female															
6	5,01/01/2023,Wed,USA,washington,38.9072,-77.0369,Sami Altawil,Headphones,Sennheiser HD,Sennheiser HD 450BT,Q149,4,129,11.2617,111.3657,True,Female															
7	6,01/01/2023,Mon,Syria,aleppo,36.2021,37.1343,Ahed Salim,Smartwatch,Garmin Fenix,Garmin Fenix 6S,J431,2,499,232.1847,138.3228,True,Male															
8	7,01/01/2023,Tue,Saudi Arabia,riyadh,24.7136,46.6753,Amira Alrahil,Digital Camera,Panasonic Lumix,Panasonic Lumix S1H,S190,3,3499,2203.6702,197.3436,True,Female															
9	8,01/01/2023,Mon,Egypt,cairo,30.0444,31.2357,Muhamad Bitahish,Headphones,Anker Soundcore,Anker Soundcore Liberty Air 2 Pro,R389,3,99,74.4876,47.0448,False,Female															
10	9,01/01/2023,Tue,Saudi Arabia,aseer,18.2311,42.5004,Fadi Aljabaan,Laptop,HP Envy,Envy x360,I974,4,899,359.6,881.02,True,Male															
11	10,01/01/2023,Mon,USA,washington,38.9072,-77.0369,Zahir Almunajid,Smart Speaker,Apple HomePod,Apple HomePod mini,R236,3,99,598.3956,672.1704,True,Female															
12	11,01/01/2023,Tue,Egypt,cairo,30.0444,31.2357,Eala Aljabaan,Laptop,Lenovo Legion,Lenovo Legion 5,B541,4,1099,1351.77,1945.23,True,Female															
13	12,01/01/2023,Tue,Egypt,cairo,30.0444,31.2357,Abd Allatif Alghandur,Tablet,Samsung Galaxy Tab,Galaxy Tab S7,U983,4,649,52.569,578.259,True,Male															
14	13,01/01/2023,Tue,USA,las vegas,36.1699,-115.1398,Muhamad Eurman,Smartwatch,Fossil Gen,Fossil Gen 5E,N710,4,199,136.116,130.743,True,Female															
15	14,01/01/2023,Mon,Morocco,casablanca,33.5731,7.5898,Shahad Eyd Bndqjy,Smartwatch,Garmin Fenix,Garmin Fenix 7,G033,5,699,388.0149,566.1201,True,Female															
16	15,01/01/2023,Mon,Egypt,asyut,27.1809,31.1837,Sahar Kharasah,Tablet,Amazon Fire,Amazon Fire 7,G449,2,49,8.6583,32.8104,True,Male															
17	16,01/01/2023,Tue,United Arab Emirates,abu dhabi,24.4539,54.3773,Muhamad Zawal,Tablet,Samsung Galaxy Tab,Galaxy Tab A7,T351,4,179,115.276,159.7396,True,Female															
18	17,01/01/2023,Mon,France,paris,48.8566,2.3522,Muhamad Zaetar,Headphones,Sennheiser HD,Sennheiser HD 660S,S691,8,499,3719.6458,4573.335,True,Female															
19	18,01/01/2023,Tue,Saudi Arabia,jeddah,21.5433,39.1728,Muhamad Byd,Tablet,Lenovo Tab,Lenovo Tab M10 FHD Plus,D241,4,229,102.2256,187.4136,True,Male															
20	19,01/01/2023,Mon,Saudi Arabia,riyadh,24.7136,46.6753,Fadi Ahmad,VR Headset,Oculus Rift,Oculus Rift S,O247,4,399,54.7428,367.5588,False,Female															
21	20,01/01/2023,Tue,Saudi Arabia,riyadh,24.7743,46.7386,Khalid Dbs,Headphones,Anker Soundcore,Anker Soundcore Life Q35,R252,4,129,299.6154,326.1636,False,Female															
22	21,01/01/2023,Tue,Saudi Arabia,aseer,18.2311,42.5004,Akthum Haydar,Digital Camera,Sony Cyber-shot,Sony Cyber-shot RX100 VII,S783,4,1199,322.2912,138.1248,True,Male															
23	22,01/01/2023,Mon,USA,new york,40.7498,-73.9814,Muhamad Zaqzuq,Headphones,JBL Live,JBL Live 500BT,J345,7,99,38.61,9.9,False,Female															
24	23,01/01/2023,Wed,United Arab Emirates,abu dhabi,24.4764,54.3705,Rwbyna Salhany,Headphones,Sony WH,Sony WH-CH710N,H632,4,129,47.73,28.38,True,Female															
25	24,01/01/2023,Tue,USA,new york,40.7498,-73.9814,Muwmin Alsaman,Headphones,Anker Soundcore,Anker Soundcore Liberty Air Pro,Q832,3,99,68.8644,22.3344,True,Male															
26	25,01/01/2023,Tue,United Arab Emirates,abu dhabi,24.4539,54.3773,Jwny Alrifaei,Laptop,Lenovo Legion,Lenovo Legion 5,P653,4,1099,652.806,1142.4105,True,Female															
27	26,01/01/2023,Tue,Syria,damascus,33.5146,36.3084,Ali Iismaeil,Laptop,Apple MacBook,"MacBook Air 13""",U470,3,999,349.2504,165.4344,False,Female															
28	27,01/01/2023,Tue,Egypt,cairo,30.0444,31.2357,Husam Aldiyn Allaham,Tablet,Amazon Fire,Amazon Fire HD 10,J431,4,149,85.9134,48.4995,True,Male															

Online Electronic Sales

يحتوي الملف كما نرى الأعمدة التالية:

- **Order ID** : رقم فريد يميز كل طلبية.
- **Order Date** : التاريخ الذي تم فيه تقديم الطلبية.
- **Day** : يوم الأسبوع الذي تم فيه تقديم الطلبية.
- **Country** : بلد المشتري.
- **City** : مدينة المشتري.
- **Lat** : خط عرض مدينة المشتري.
- **Lng** : خط طول مدينة المشتري.
- **Full Name** : الاسم الكامل للمشتري.
- **Category** : فئة المنتج الذي تم شراؤه.
- **Sub Category** : فئة فرعية للمنتج الذي تم شراؤه.
- **Item** : اسم المنتج الذي تم شراؤه.
- **Sales Person ID** : معرف مندوب المبيعات الذي باع المنتج.
- **Quantity** : كمية المنتج التي تم شراؤها.
- **Unit Price** : سعر الوحدة للمنتج.
- **Discount** : خصم على سعر المنتج.
- **Total Cost** : التكلفة الإجمالية للطلبية، وهي حاصل ضرب كمية المنتج في سعر الوحدة مع خصم.
- **Status** : حالة الطلبية، ويمكن أن تكون "False" للطلبات التي لم يتم شحنها بعد، أو "True" للطلبات التي تم شحنها.
- **Gender** : جنس المشتري، ويمكن أن يكون "Male" للذكور أو "Female" للإناث.

المكتبات المستخدمة لإنشاء المخططات:

1. Pandas (pd):

- **الاسم الكامل:** Pandas
- **الغرض:** مكتبة معالجة وتحليل البيانات.

- **المكونات الأساسية:**
  - **Series**: سلسلة بيانات أحادية البعد مع تسميات.
  - **DataFrame**: جدول بيانات ثنائي الأبعاد مع تسميات للصفوف والأعمدة.
  - **Data Manipulation**: أدوات لتنظيف البيانات وتحويلها وتحليلها.
  - **Time Series Analysis**: أدوات لتحليل البيانات المتسلسلة.
- **المكونات الإضافية:**
  - **Index:** بنية بيانات لتخزين مفاتيح فريدة للوصول إلى البيانات.
  - **MultiIndex**: بنية بيانات لتخزين مفاتيح متعددة الأبعاد للوصول إلى البيانات.
  - **Data Structures**: هياكل بيانات مثل المصفوفات والقواميس لتخزين البيانات.
  - **Plotting**: أدوات بسيطة لرسم البيانات.
- **الميزات :**
  - تحميل البيانات من مصادر مختلفة (CSV، إكسل، قواعد البيانات).
  - إنشاء جداول البيانات (DataFrames) ومعالجتها.
  - تنظيف البيانات وتحويلها (معالجة القيم المفقودة، والتصفية، والتجميع).
  - إجراء العمليات الحسابية والتجميعية على البيانات.
- **حالات الاستخدام:** تنظيف البيانات وتحضيرها وتحليلها لمهام متنوعة مثل التعلم الآلي، التحليل المالي، والحساب العلمي.

2. matplotlib. pyplot (plt):

- **الاسم الكامل:** Matplotlib (وحدة pyplot الفرعية)

- **الغرض:** إنشاء تصورات ثابتة مثل الرسوم البيانية الخطية، ومخططات النقاط، والمخططات الشريطية.
- **المكونات الأساسية:**
  - Figures: تمثل مساحة الرسم للمخططات.
  - Axes: تمثل مساحة الرسم الفعلية للمخططات داخل الشكل.
  - Lines: تُستخدم لرسم خطوط على المخططات.
  - Plots: تُستخدم لإنشاء أنواع مختلفة من المخططات (خطية، شريطية، دائرية، إلخ).
- **المكونات الإضافية:**
  - Annotations: إضافة تعليقات ونصوص للمخططات.
  - Legends: إنشاء أساطير لشرح العناصر في المخطط.
  - Subplots: إنشاء مخططات فرعية متعددة في نفس الشكل.
  - Styling: أدوات لتخصيص مظهر المخططات (ألوان، خطوط، علامات، إلخ).
- **الميزات :**
  - تشكيلة واسعة من أنواع الرسوم والتخصيصات.
  - التحكم في عناصر الرسم (المحاور، العلامات، التسميات التوضيحية).
  - التكامل مع المكتبات الأخرى (NumPy، pandas).
- **حالات الاستخدام:** إنشاء رسوم بيانية عالية الجودة للنشر، ورسومات تحليلية استكشافية للبيانات، واستكشاف البيانات الأساسي.

3. Seaborn (sns):

- **الاسم الكامل:** Seaborn
- **الغرض:** واجهة عالية المستوى مبنية على Matplotlib لإنشاء رسومات بيانية إحصائية.
- **المكونات الأساسية:**
  - Statistical Functions: وظائف إحصائية متقدمة لتحليل البيانات.
  - Data Visualization: أدوات متقدمة لرسم البيانات بجودة عالية وتفاعلية.
  - Themes: مواضيع جاهزة لتخصيص مظهر المخططات بسهولة.
  - Statistical Plots: أنواع متقدمة من المخططات الإحصائية (كثافة، توزيع، ارتباط، إلخ).

- **المكونات الإضافية:**
  - Faceting: تقسيم المخططات حسب متغيرات متعددة.
  - Grids: إنشاء شبكات من المخططات الفرعية.
  - Statistical Tests: إجراء اختبارات إحصائية وتفسير النتائج.
  - Interactive Plots: إنشاء مخططات تفاعلية قابلة للتعديل.
- **الميزات :**
  - رسومات بيانية جذابة وغنية بالمعلومات.
  - سمات وأساليب محددة مسبقاً للرسومات المتسقة.
  - وظائف للرسومات البيانية الإحصائية المحددة (الخرائط الحرارية، ومخططات الكمان، ومخططات التوزيع).
- **حالات الاستخدام:** إنشاء رسومات بيانية إحصائية جذابة وغنية بالمعلومات للعروض التقديمية، والتقارير، واستكشاف البيانات.

4. NumPy (np):

- **الاسم الكامل:** NumPy (Numerical Python)
- **الغرض:** أساس الحساب العلمي في Python، ومعالجة البيانات العددية بكفاءة.
- **المكونات الأساسية:**
  - ndarray (n-dimensional array) : البنية الأساسية للمصفوفات متعددة الأبعاد.
  - Universal Functions (ufuncs): مجموعة واسعة من الدوال الرياضية والإحصائية.
  - Broadcasting: ميزة البث للتعامل مع مصفوفات ذات أبعاد مختلفة.
  - I/O Tools* أدوات لقراءة وكتابة البيانات من/إلى صيغ مختلفة.
- **المكونات الإضافية:**
  - Indexing and Slicing: إمكانية الوصول إلى عناصر المصفوفة باستخدام الفهرسة والشرائح.
  - Fancy Indexing: طرق متقدمة لفهرسة المصفوفات باستخدام مصفوفات.

- **الميزات :**

- صفيفات متعددة الأبعاد (ndarrays) لتخزين البيانات ومعالجتها بكفاءة.
- وظائف رياضية (الجمع، والطرح، وعمليات المصفوفات، والحسابات العنصرية).
- عمليات الجبر الخطي (حل المعادلات، وإيجاد القيم الذاتية، وتحليل المصفوفات).
- إنشاء أعداد عشوائية.

- **حالات الاستخدام:** خوارزميات التعلم الآلي، وتحليل البيانات (الحسابات العددية)، ومهام الحساب العلمي.

5. plotly. express (px):

- **الاسم الكامل:** Plotly Express
- **المكونات أساسية:**
  - Rapid Visualization Creation: تسمح Plotly Express بإنشاء رسوم بيانية تفاعلية بشكل سريع وبمجرد سطر واحد من الكود.
  - Supported Plot Types: تدعم Plotly Express مجموعة واسعة من أنواع الرسوم البيانية مثل المخططات الخطية والمبعثرة والتكراريةوغيرها.
  - Data Input: يمكن تمرير البيانات إلى Plotly Express في شكل DataFrame من Pandas أو قوائم أو np.ndarray منNumPy
  - Customization: توفر Plotly Express الكثير من الخيارات للتخصيص مثل تغيير الألوان والعناوين والتسميات وما إلى ذلك.
- **المكونات الإضافية:**
  - Plotly Figure: الرسوم البيانية المنشأة بواسطة Plotly Express هي أشياء Plotly Figureكاملة والتي تتيح المزيد من التخصيص المتقدم.
  - Plotly Dashboards: إمكانية إنشاء لوحات معلومات تفاعلية باستخدام.Plotly Dash .
  - Animations: دعم إنشاء رسوم بيانية متحركة.
  - Annotations and Interactivity: إضافة تعليقات وميزات تفاعلية للرسوم البيانية.
  - Plotly Express Functions: مجموعة واسعة من الوظائف المتخصصة مثل scatter_geo()وparallel_coordinates()
  - Plotly Themes: إمكانية تطبيق أنماط موحدة على مجموعة من الرسوم البيانية.

- **الميزات :**
  - تركيبة بسيطة لإنشاء رسوم بيانية متنوعة (مخططات النقاط، والمخططات الشريطية، والخرائط الحرارية).
  - ميزات تفاعلية مثل التكبير، والتمرير، ونصائح الأدوات.
- **حالات الاستخدام:** إنشاء تصورات بيانات تفاعلية لتطبيقات الويب، ولوحات المعلومات، واستكشاف البيانات.

6. Holoviews (hv)

- **الاسم الكامل:** HoloViews
- **الغرض:** HoloViews هي مكتبة قوية لإنشاء تصورات علمية تفاعلية في لغة البرمجة Python. تُستخدم على نطاق واسع من قبل الباحثين والعلماء ومطوري البيانات لإنشاء رسومات بيانية وتصورات غنية بالمعلومات تسهل استكشاف البيانات وفهمها.
- **المكونات الأساسية:**
  - Core Data Structures: تدعم Holoviews بنى بيانات أساسية مثل Curve و Scatter و Image وGroupBy .
  - Declarative Plotting: تسمح Holoviews للمستخدمين بتعريف الرسوم البيانية بشكل تصريحي دون الحاجة إلى كتابة الكثير من التفاصيل البرمجية.
  - Automatic Linking: تربط Holoviews تلقائياً بين الرسوم البيانية المختلفة للسماح بالتفاعل والتحليل المتكامل.
  - Customization: توفر Holoviews خصائص التخصيص مثل تغيير العنوان والمحاور والألوان والتصنيف.
  - Streaming and Interactivity
  - : تدعم Holoviews عرض البيانات التدفقية والتفاعل مع الرسوم البيانية.

- **المكونات الإضافية:**
  - Underlying Plotting Backends: تتكامل Holoviews مع مكتبات الرسم مثل Matplotlibو Plotly وBokeh .
  - Aggregation and Grouping: توفر Holoviews أدوات لتجميع وتصنيف البيانات.
  - Geographical Visualizations: تدعم خرائط جغرافية تفاعلية باستخدامGeoviews .
  - Panel: إمكانية إنشاء تطبيقات ويب تفاعلية باستخدام مكتبةPanel .
  - Datashader: مكتبة لتحسين أداء الرسوم البيانية للبيانات الكبيرة.
  - Param: نظام إعدادات لإنشاء واجهات برمجية سهلة الاستخدام.
  - Contrib Packages: مجموعة من الحزم الإضافية التي توسع إمكانيات.Holoviews
- **الميزات :**
  - **إنشاء تصورات تفاعلية:** HoloViews تسمح بإنشاء تصورات تفاعلية يمكن للمستخدمين التفاعل معها لتغيير منظورهم على البيانات واستكشافها بشكل أعمق.
  - **تركيز على البيانات:** تركز HoloViews على البيانات نفسها، مما يسهل على المستخدمين تصميم تصورات تناسب احتياجاتهم الخاصة وتحليل البيانات.
  - **تنسيقات مرنة:** تدعم HoloViews مجموعة متنوعة من تنسيقات البيانات، بما في ذلك DataFrames و Arrays و Dictionaries.
  - **دمج مع مكتبات أخرى:** تتكامل بسلاسة مع مكتبات Python الأخرى مثل pandas و NumPy و Matplotlib و Bokeh، مما يوفر بيئة عمل متكاملة لتحليل البيانات وتصورها.
- **حالات الاستخدام:**
  - **استكشاف البيانات:** HoloViews تُستخدم بشكل فعال لاستكشاف مجموعات البيانات الكبيرة ومعرفة الأنماط والعلاقات المخفية فيها.
  - **التحليل العلمي:** HoloViews تُستخدم من قبل الباحثين والعلماء لإنشاء تصورات تفاعلية تُساعدهم على فهم نتائج أبحاثهم بشكل أفضل.
  - **تطوير التطبيقات:** HoloViews تُستخدم في تطوير تطبيقات تحليل البيانات وتصورها، مما يسمح للمستخدمين بالتفاعل مع البيانات بشكل مباشر.
  - **التواصل العلمي:** HoloViews تُستخدم لإنشاء تصورات جذابة وفعالة لتوصيل النتائج العلمية للآخرين.

معاينة البيانات:

نبدأ أولا بتحميل البيانات في إطار البيانات ومعرفة شكل البيانات ومعلومات عنها :

1.تحميل البيانات في إطار البيانات باستخدام مكتبة Pandas:

```
# Read the data from the CSV file
df = pd.read_csv(r"C:\Users\LEGION\Desktop\python\Online Electronic Sales.csv")
df
```

	Order ID	Order Date	Day	Country	City	Lat	Lng	Full Name	Category	Sub Category	Item	SalesPerson ID	Quantity	Unit Price	Discount	Total Cost	Status	Gender
0	1	01/01/2023	Mon	Syria	homs	34.7326	36.7136	Lina Alrrashid	Tablet	Apple iPad	iPad Pro 12.9"	N498	4	999	38.3616	891.9072	False	Female
1	2	01/01/2023	Tue	Saudi Arabia	riyadh	24.7136	46.6753	Omar Eurul	Smartphone	Samsung Galaxy	Galaxy S21 Ultra	X918	3	1199	517.9680	302.1480	True	Female
2	3	01/01/2023	Tue	Saudi Arabia	riyadh	24.7743	46.7386	Iman Ismaeil	Digital Camera	Panasonic Lumix	Panasonic Lumix GH5	I036	4	1299	883.3200	831.3600	True	Male
3	4	01/01/2023	Mon	United Arab Emirates	abu dhabi	24.4539	54.3773	Ahmad Rihan	Tablet	Samsung Galaxy Tab	Galaxy Tab A8	E804	6	199	33.3126	129.5490	True	Female
4	5	01/01/2023	Wed	USA	washington	38.9072	-77.0369	Sami Altawil	Headphones	Sennheiser HD	Sennheiser HD 450BT	Q149	4	129	11.2617	111.3657	True	Female
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
19995	19996	01/09/2023	Tue	Morocco	casablanca	33.5731	7.5898	Ahmad Iad	Gaming Console	Sony PlayStation	PlayStation 4 Pro	M210	4	399	150.9417	178.0338	True	Female
19996	19997	01/09/2023	Tue	Syria	homs	34.7326	36.7136	Ali Kiali	Smartwatch	Fossil Gen	Fossil Gen 6	Z826	3	299	398.2680	215.2800	True	Female
19997	19998	01/09/2023	Wed	USA	las vegas	36.1699	-115.1398	Husayn Salayk	Gaming Console	Sony PlayStation	PlayStation 2	K624	5	99	15.3648	56.6577	True	Male
19998	19999	01/09/2023	Mon	Saudi Arabia	jeddah	21.4858	39.1925	Fatin Bahriin	Headphones	JBL Live	JBL Live 500BT	Y368	4	99	19.9584	68.4288	True	Female

2. شكل البيانات باستخدام دالة (df.shape):

```
df.shape
```

```
(20000, 18)
```

مما يظهر أن البيانات تتألف من 2000 صفاً و13 عموداً

3.معلومات البيانات:

يمكن استخدام دالة المعلومات Info لمعاينة أنماط بيانات الأعمدة وفيما إذا يوجد قيم فارغة.

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 20000 entries, 0 to 19999
Data columns (total 18 columns):
 #   Column          Non-Null Count  Dtype
---  ------          --------------  -----
 0   Order ID        20000 non-null  int64
 1   Order Date      20000 non-null  object
 2   Day             20000 non-null  object
 3   Country         20000 non-null  object
 4   City            20000 non-null  object
 5   Lat             20000 non-null  float64
 6   Lng             20000 non-null  float64
 7   Full Name       20000 non-null  object
 8   Category        20000 non-null  object
 9   Sub Category    20000 non-null  object
 10  Item            20000 non-null  object
 11  SalesPerson ID  20000 non-null  object
 12  Quantity        20000 non-null  int64
 13  Unit Price      20000 non-null  int64
 14  Discount        20000 non-null  float64
 15  Total Cost      20000 non-null  float64
 16  Status          20000 non-null  bool
 17  Gender          20000 non-null  object
dtypes: bool(1), float64(4), int64(3), object(10)
memory usage: 2.6+ MB
```

مما يظهر أن لجميع الأعمدة 2000 قيمة غير فارغة non-null. كما أنه يوجد أعمدة رقمية صحيحة int64 وأعمدة غير رقمية object وأعمدة رقمية حقيقة float64 وعمود قيمه منطقية bool.

إنشاء المخططات العشرة:

- المخطط الشريطي العمودي:

الفكرة الأساسية من هذا المخطط هي إظهار عدد الناس من كل بلد في مجموعة البيانات باستخدام مكتبة matplotlib. pyplot.

```
# Count the number of people from each country
country_counts = df['Country'].value_counts()

# Define a list of colors for the bars (up to 7 countries)
colors = ['r', 'g', 'b', 'c', 'm', 'y', 'k']

# Create a bar chart to visualize the number of people per country
country_counts.plot(kind='bar', figsize=(10, 6), color=colors)

# Add labels to the x and y axes
plt.xlabel('Country')
plt.ylabel('Number of People')

# Add a title to the chart
plt.title('Number of People Per Country')

# Rotate x-axis labels for better readability (if many countries)
plt.xticks(rotation=45)

# Display the chart
plt.show()
```

أهم الباراميترات في الرماز الخاص بالمخطط الشريطي العمودي:

1. **استيراد المكتبات:**
   - `pandas`: لمعالجة البيانات وتحليلها.
   - `matplotlib. pyplot`: لإنشاء عناصر الرسم البياني الأساسية.
2. **عدّ الأشخاص لكل بلد:**
   - نستخدم `df['Country'].value_counts()` لإنشاء سلسلة `country_counts` تُظهر عدد كل بلد فريد.
3. **تحديد ألوان الأعمدة:**
   - أنشئ قائمة `colors` تحتوي على أكواد ألوان لعدد 7 أعمدة كحد أقصى.
4. **إنشاء مخطط شريطي:**

- `country_counts. plot (kind='bar', figsize= (10, 6), color=colors)` لإنشاء مخطط شريطي.
  - `kind='bar'`: يحدد نوع المخطط.
  - `figsize= (10، 6)`: يضبط حجم الرسم.
  - `color=colors`: يعين ألوان الأعمدة من قائمة `colors`.

5. **إضافة تسميات المحاور:**
   - `plt. xlabel('Country')`: يضيف تسمية "بلد" لمحور x.
   - `plt. ylabel ('Number of People')`: يضيف تسمية "عدد الأشخاص" لمحور y.
6. **إضافة عنوان للمخطط:**
   - `plt. title ('Number of People Per Country')`: يضيف عنوان "عدد الأشخاص لكل بلد" للمخطط.
7. **تدوير تسميات محور x (اختياري):**
   - `plt.xticks(rotation=45)`: يدير تسميات محور x بزاوية 45 درجة لتحسين القراءة (في حال وجود العديد من البلدان).
8. **عرض المخطط:**
   - `plt.show()`: يعرض المخطط المُنشأ.

مخرجات الرماز البرمجي الخاص بالمخطط الشريطي العمودي:

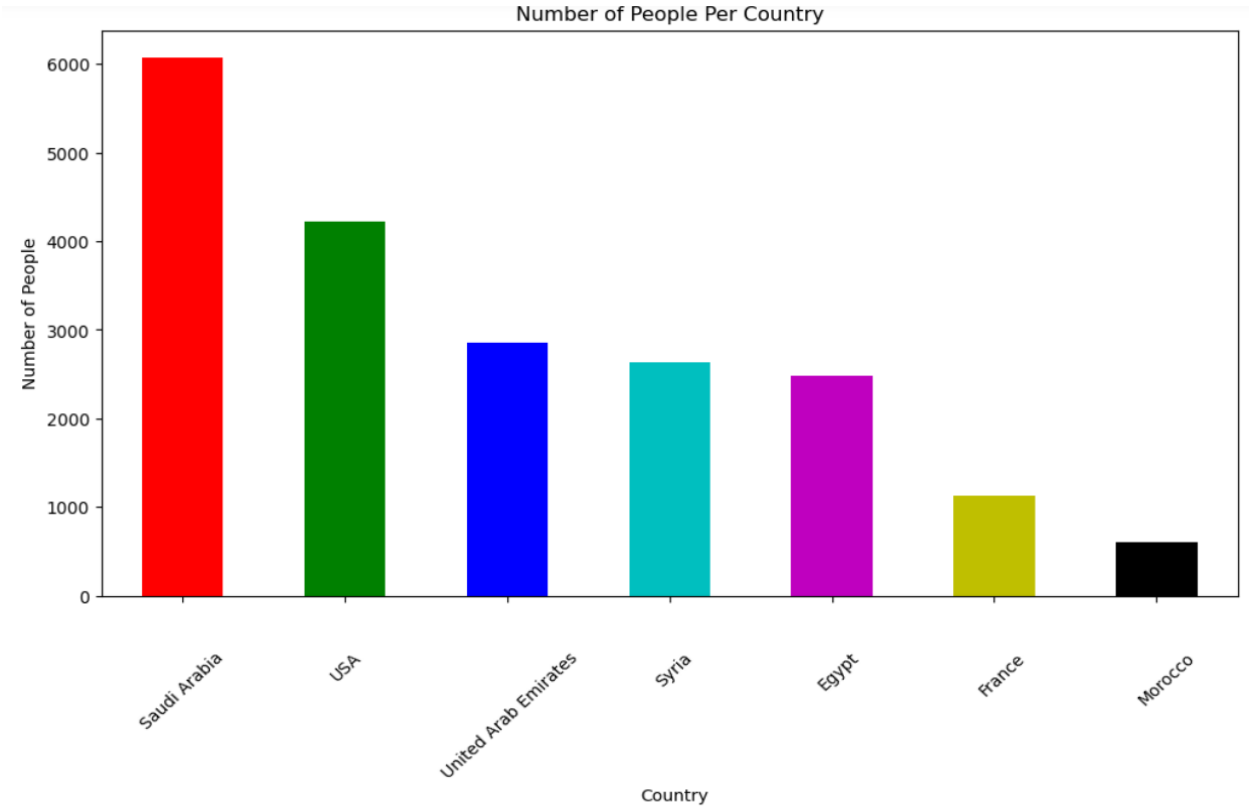
من هذا المخطط البياني يمكننا استنتاج ما يلي بشكل مختصر:

1 .توزيع الأشخاص حسب الدول: المخطط يُظهر عدد الأشخاص من كل دولة في البيانات، مما يسمح بمقارنة الدول من حيث التمثيل.

2 .التركيز على الدول الرئيسية: يمكن الاهتمام بالدول ذات العدد الأكبر من الأشخاص واستكشاف العوامل المؤثرة على ذلك.

3 .إمكانية التخطيط والتسويق: هذه المعلومات يمكن استخدامها في التخطيط والتسويق بشكل أفضل، مثل توجيه الموارد والجهود نحو الدول الأكثر تمثيلاً.

4 .التحليل المقارن: المخطط يسمح بمقارنة الدول من حيث عدد الأشخاص، مما قد يكشف عن اتجاهات أو فرص جديدة.


Number of People Per Country
Number of People
0
1000
2000
3000
4000
5000
6000
Saudi Arabia
USA
United Arab Emirates
Syria
Egypt
France
Morocco
Country

- **المخطط الشريطي الافقي:**

الفكرة الرئيسة وراء هذا المخطط هي إظهار إجمالي التكلفة لكل فئة في مجموعة البيانات بطريقة بصرية وسهلة الفهم باستخدام مكتبة plotly. express.

```
# Group data by 'Category' and calculate total cost for each category
category_totals = df.groupby('Category')['Total Cost'].sum().reset_index()

# Create a horizontal bar chart using plotly.express
fig = px.bar(category_totals, x='Total Cost', y='Category', orientation='h',
             title='Total Cost by Category', color='Category')

# Display the chart
fig.show()
```

أهم الباراميترات في الرماز البرمجي الخاص بالمخطط الشريطي الافقي:

1. **استيراد المكتبات :**
   - s: لمعالجة البيانات وتحليلها.
   - plotly. express: لإنشاء رسومات تفاعلية باستخدام مكتبة Plotly.

1. **حساب إجمالي التكلفة حسب الفئة:**
   - df.groupby('Category')['Total Cost'].sum().reset_index()=category_totals
     - يُجمّع البيانات في إطار البيانات df حسب عمود "الفئة".
     - يحسب إجمالي قيمة "إجمالي التكلفة" لكل فئة باستخدام sum().
     - يعيد ضبط الفهرس باستخدام() reset_index لتحويل المجموعات إلى أعمدة.
     - يخزن النتيجة في متغير جديد category_totals.
2. **إنشاء مخطط أعمدة أفقي:**
   - fig = px.bar(...):
     - يستخدم px.bar من مكتبة Plotly Express لإنشاء مخطط أعمدة.
     - يُمرر category_totals كحجة، وهي تحتوي على إجمالي التكلفة لكل فئة.
3. **تخصيص المخطط:**
   - x='Total Cost’:
     - يُعيّن قيم المحور الأفقي (x) بقيم "إجمالي التكلفة".
   - y='Category’ :

- يُعيّن قيم المحور العمودي (y) بأسماء الفئات (فهرس `category_totals`).

o `orientation='h'`:

- يُغيّر اتجاه المخطط إلى أفقي باستخدام `orientation='h'`.

o `title='Total Cost by Category'`:

- يُضيف عنوان رئيسي للمخطط "إجمالي التكلفة حسب الفئة" باستخدام `title`.

o `color='Category'`:

- يُستخدم عمود "الفئة" لتحديد لون كل عمود في المخطط.

4. **عرض المخطط:**

o `fig.show()`:

- يعرض مخطط الأعمدة الأفقي المُنشأ باستخدام `fig.show`.

مخرجات الرماز البرمجي الخاص بالمخطط الشريطي الافقي:

من هذا المخطط البياني يمكننا استنتاج ما يلي بشكل مختصر:

1 .التوزيع حسب الفئات: المخطط يُظهر إجمالي التكلفة لكل فئة من الفئات الموجودة في البيانات.

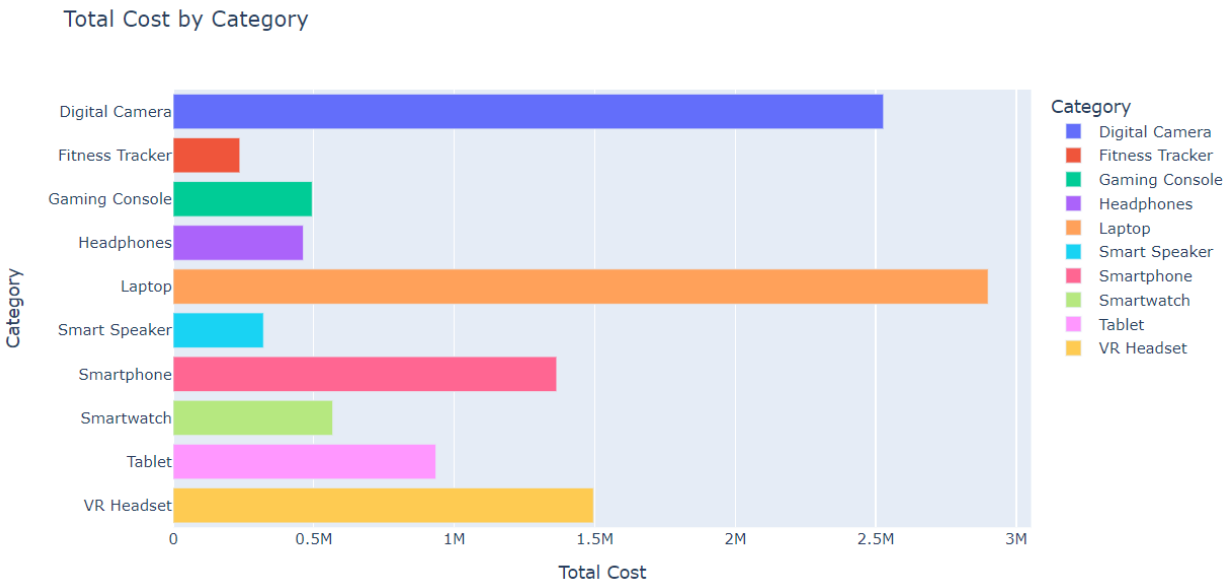
2 .القيم المطلقة: نلاحظ القيم المطلقة للتكلفة الإجمالية لكل فئة، مما يساعد في فهم الأهمية النسبية لكل منها.

3 .التركيز على الفئات الرئيسية: يمكن التركيز على الفئات ذات التكاليف الإجمالية الأعلى واستكشاف أسباب ذلك وإمكانية تحسينها.

4 .التخطيط والموازنة: هذه المعلومات يمكن استخدامها في التخطيط والموازنة، مثل تخصيص الموارد بشكل أفضل بين الفئات المختلفة.

5 .المقارنة عبر الوقت: إذا تم عمل هذا المخطط بشكل دوري، فيمكن مراقبة التغيرات في توزيع التكاليف على الفئات عبر الوقت.

6 .البساطة والوضوح: استخدام المخطط البياني الأفقي يجعل العرض بسيطاً وسهل الفهم.


Total Cost by Category
Digital Camera
Fitness Tracker
Gaming Console
Headphones
Laptop
Smart Speaker
Smartphone
Smartwatch
Tablet
VR Headset
Category
0
0.5M
1M
1.5M
2M
2.5M
3M
Total Cost
Category

- **المخطط الدائري:**

الفكرة الرئيسية من هذا المخطط هي توضيح توزيع الطلبات حسب الحالات المختلفة (قيد التنفيذ، مكتمل، ملغي، إلخ.) باستخدام مكتبة matplotlib.pyplot.

```
# Count the number of orders for each status
status_counts = df['Status'].value_counts()

# Create a figure for the pie chart with a specific size
plt.figure(figsize=(4, 4))

# Create a pie chart with labels, percentages, and customization
plt.pie(status_counts,
        labels=status_counts.index,  # Use status names as labels
        autopct='%1.1f%%',           # Display percentages with one decimal place
        startangle=90,              # Start the pie at 90 degrees (optional)
        colors=['#9AC8CD', '#E1F7F5'])  # Set custom colors

# Add a title for the chart
plt.title('Orders By Status')

# Ensure the pie chart is a circle
plt.axis('equal')

# Display the pie chart
plt.show()
```

أهم الباراميترات في الرماز البرمجي الخاص بالمخطط الدائري:

1. **استيراد المكتبات:**
   - `pandas`: لمعالجة البيانات وتحليلها.
   - `matplotlib. pyplot`: لإنشاء عناصر الرسم البياني الأساسية.
2. **عدّ الطلبات حسب الحالة:**
   - نستخدم `df['Status'].value_counts ()` لإنشاء سلسلة `status_counts` تُظهر عدد الطلبات لكل حالة فريدة.

3. **إنشاء مخطط دائري:**
   - `plt.pie(status_counts,labels=status_counts.index, autopct='%1.1f%%', startangle=90, colors=['#9AC8CD', '#E1F7F5'])` لإنشاء مخطط دائري باستخدام `matplotlib.pyplot`.
     - `status_counts`: سلسلة تحتوي على عدد الطلبات لكل حالة.
     - `labels=status_counts.index`: يُستخدم أسماء الحالات (فهرس `status_counts`) كملصقات للمخطط.
     - `autopct='%1.1f%%'`: يعرض النسب المئوية بجانب كل شريحة مع رقم عشري واحد (مثل "10.5٪").
     - `startangle=90`: يضبط زاوية بدء المخطط الدائري إلى 90 درجة (اختياري).
     - `colors=['#9AC8CD', '#E1F7F5']`: يحدد لونين مخصصين لشرائح المخطط الدائري.
4. **إضافة عنوان المخطط:**
   - نستخدم `plt.title('Orders By Status')` لإضافة عنوان "طلبات حسب الحالة" للمخطط.
5. **ضمان نسبة العرض إلى الارتفاع:**
   - نستخدم `plt.axis('equal')` لضمان رسم المخطط الدائري كدائرة دون تشويه.
6. **عرض المخطط:**
   - نستخدم `plt.show()` لعرض المخطط المُنشأ.

مخرجات الرماز البرمجي الخاص بالمخطط الدائري:

من هذا المخطط البياني يمكننا استنتاج ما يلي بشكل مختصر:

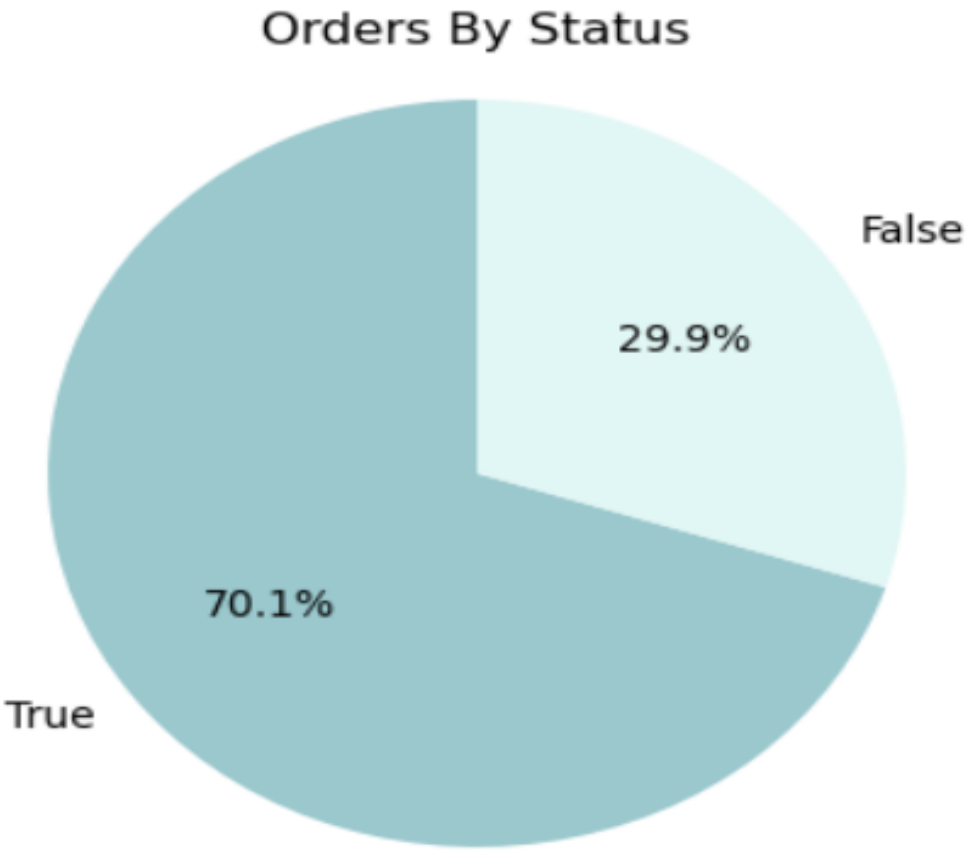
1. التوزيع حسب حالة الطلبات: المخطط يُظهر توزيع الطلبات على الحالات المختلفة (مثل المكتملة، قيد المعالجة، ملغاة، إلخ).

2. النسب المئوية: نلاحظ النسب المئوية لكل حالة، مما يساعد في فهم أهمية نسبية لكل منها.

3. التركيز على الحالات الرئيسية: يمكن التركيز على الحالات ذات النسبة الأعلى واستكشاف أسباب ذلك والعمل على تحسينها.

4. التخطيط والرقابة: هذه المعلومات يمكن استخدامها في التخطيط والرقابة على العمليات، مثل تحديد الأهداف لتحسين الحالات الأقل نسبة.

5. المقارنة عبر الوقت: إذا تم عمل هذا المخطط بشكل دوري، فيمكن مراقبة التغيرات في توزيع الحالات عبر الوقت.


Orders By Status
False
29.9%
70.1%
True

- المخطط الشريطي المقارن:

الفكرة الأساسية من إنشاء هذا المخطط باستخدام مكتبة plotly. express هي عرض عدد الذكور والإناث في كل دولة.

```
# Group data by 'Country' and 'Gender', then count occurrences
country_gender_counts = df.groupby(['Country', 'Gender']).size().reset_index(name='Count')

# Create a grouped bar chart using plotly.express
fig = px.bar(country_gender_counts, x='Country', y='Count', color='Gender',
             barmode='group', title='Number of Males and Females in Each Country')

# Display the chart
fig.show()
```

أهم الباراميترات في الرماز البرمجي الخاص بالمخطط الشريطي المقارن:

1. **استيراد المكتبات:**
   - `s`: لمعالجة البيانات وتحليلها.
   - `plotly. express`: لإنشاء رسومات تفاعلية باستخدام مكتبة Plotly.

1. **عد الأفراد حسب الدولة والجنس:**
   - `country_gender_counts = df.groupby(['Country', 'Gender']).size().reset_index(name='Count')`:
     - يُجمّع البيانات في إطار البيانات `df` حسب كل من "الدولة" و"الجنس".
     - يستخدم `size()` لحساب عدد مرات ظهور كل مجموعة (عدد الأفراد لكل دولة وجنس).
     - يعيد ضبط الفهرس باستخدام `reset_index()` لتحويل المجموعات إلى أعمدة.
     - يُسمى العمود الجديد الذي يحمل قيم العد بـ "Count".
     - يخزن النتيجة في متغير جديد `country_gender_counts`.
2. **إنشاء مخطط أعمدة مقارنة:**
   - `fig = px.bar(...)`:
     - يستخدم `px.bar` من مكتبة Plotly Express لإنشاء مخطط أعمدة.
     - يُمرر `country_gender_counts` كحجة، وهي تحتوي على عدد الأفراد لكل مجموعة (دولة وجنس).

3. **تخصيص المخطط:**
   - `'x='Country :`
     - يُعيّن قيم المحور الأفقي (x) بأسماء الدول (عمود "Country").
   - `'y='Count :`
     - يُعيّن قيم المحور العمودي (y) بقيم عدد الأفراد (عمود "Count").
   - `'color='Gender :`
     - يُستخدم عمود "الجنس" لتحديد لون كل مجموعة أعمدة في المخطط (ذكورا أو إناث).
   - `'barmode='group :`
     - يُحدد نوع المخطط بـ "group" لإنشاء أعمدة مقارنة (مكدسة فوق بعضها).
   - `'title='Number of Males and Females in Each Country :`
     - يُضيف عنوان رئيسي للمخطط "عدد الذكور والإناث في كل دولة" باستخدام `title`.
4. **عرض المخطط:**
   - `fig.show()`:
     - يعرض مخطط الأعمدة المقارن المُنشأ باستخدام `fig.show`.

مخرجات الرماز البرمجي الخاص بالمخطط الشريطي المقارن:

من هذا المخطط البياني يمكننا استنتاج ما يلي بشكل مختصر:

1 .التوزيع حسب الدول والجنس: المخطط يُظهر عدد الذكور والإناث في كل دولة على حدة.

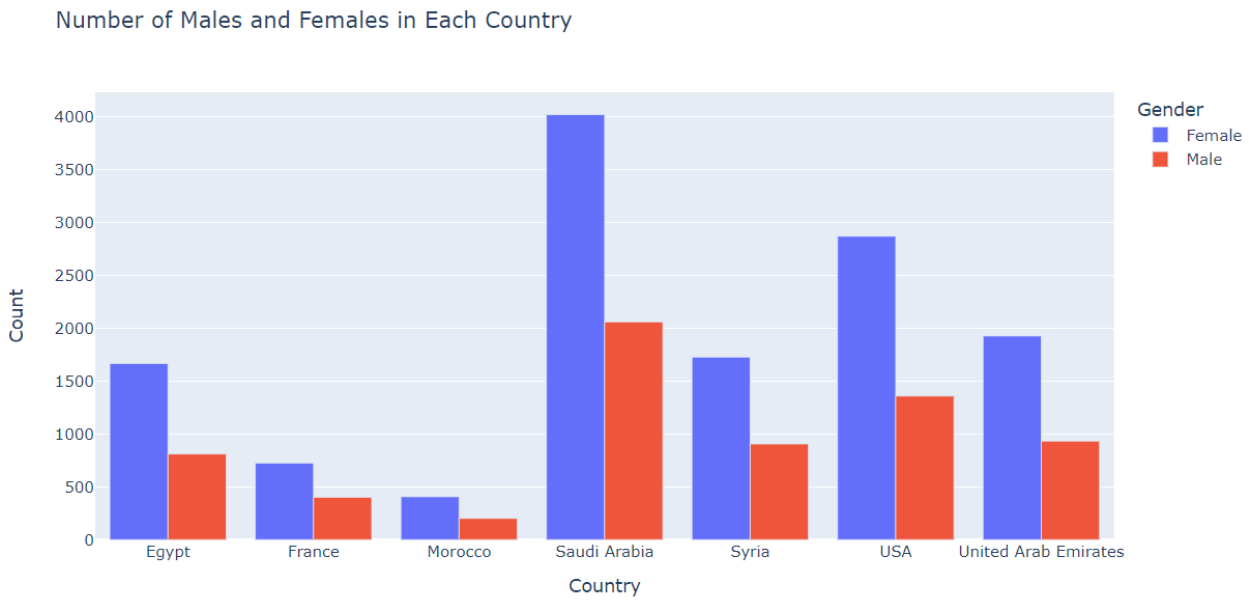
2 .المقارنة بين الجنسين: يمكن مقارنة عدد الذكور والإناث في كل دولة، وهذا يساعد في تحديد التوزيعات النسبية بين الجنسين في كل دولة.

3 .الاختلافات بين الدول: المخطط يُظهر الاختلافات في أعداد الذكور والإناث بين الدول المختلفة، مما يساعد في فهم التركيبة السكانية لكل دولة.

4 .التحليل النسبي: من خلال مقارنة أعداد الذكور والإناث في كل دولة، يمكن استخراج النسب المئوية لكل جنس ضمن كل دولة.

5 .دعم اتخاذ القرارات: هذه المعلومات قد تكون مفيدة في اتخاذ قرارات تتعلق بالسياسات والبرامج المستهدفة لكل جنس في كل دولة.

6 .البساطة والوضوح: استخدام المخطط المجمع يجعل العرض بسيطًا وسهل الفهم، خاصةً عند وجود بيانات متعددة للمقارنة.



- **المخطط الشلالي Waterfall:**

الفكرة الأساسية من هذا المخطط استخدام مكتبة matplotlib.pyplot لعرض إجمالي التكلفة في كل مدينة، بالإضافة إلى النسبة المئوية من إجمالي التكلفة لكل مدينة.

```
# Group data by 'City' and calculate total cost for each city
city_totals = df.groupby('City')['Total Cost'].sum().reset_index()
# Sort cities by total cost (highest to lowest)
city_totals = city_totals.sort_values(by='Total Cost', ascending=False)
# Calculate total cost across all cities
total = city_totals['Total Cost'].sum()
# Calculate percentage of total cost for each city
city_totals['Percentage'] = (city_totals['Total Cost'] / total) * 100
# Create a figure and an axis for the chart
fig, ax = plt.subplots(figsize=(10, 6))
# Initialize a variable to track the bottom position for stacked bars
bottom = 0

# Iterate through each row in the city_totals DataFrame
for i, row in city_totals.iterrows():
    # Create a bar for each city with its total cost and label (including percentage)
    ax.bar(row['City'], row['Total Cost'], bottom=bottom, label=f"{row['City']} ({row['Percentage']:.2f}%)")
    # Update the bottom position for the next bar based on the current city's total cost
    bottom += row['Total Cost']
# Set labels for the axes and title for the chart
ax.set_xlabel('City')
ax.set_ylabel('Total Cost')
ax.set_title('Total Cost By City (Waterfall Chart)')
# Add a legend to identify bars with city names and percentages
ax.legend(loc='best')

# Rotate x-axis labels for better readability (if many cities)
plt.xticks(rotation=45)

# Display the waterfall chart
plt.show()
```

أهم الباراميترات في الرماز البرمجي الخاص بالمخطط الشلالي:

1. **استيراد المكتبات:**
   - `pandas`: لمعالجة البيانات وتحليلها.
   - `matplotlib. pyplot`: لإنشاء عناصر الرسم البياني الأساسية.
2. **حساب إجمالي التكاليف لكل مدينة:**
   - ستخدام `df.groupby('City')['Total Cost'].sum().reset_index()` لإنشاء DataFrame جديد `city_totals` يُظهر إجمالي "التكلفة الكلية" لكل مدينة.
     - `groupby('City')`: يُجمع البيانات حسب عمود "المدينة".

- `['Total Cost'].sum()`: يحسب مجموع "التكلفة الكلية" لكل مدينة.
- `reset_index()`: يُعيد ضبط مؤشرات DataFrame لاحتواء "المدينة" كعمود و "التكلفة الكلية" كعمود جديد.

3. **فرز المدن حسب التكلفة الكلية:**
   - `city_totals.sort_values(by='Total Cost', ascending=False)` لفرز DataFrame `city_totals` حسب "التكلفة الكلية" (من الأعلى إلى الأدنى).
4. **حساب إجمالي التكاليف:**
   - استخدام `total = city_totals['Total Cost'].sum()` لحساب إجمالي "التكلفة الكلية" عبر جميع المدن.
5. **حساب النسبة المئوية:**
   - استخدام `city_totals['Percentage'] = (city_totals['Total Cost'] / total) * 100` لحساب "النسبة المئوية" من إجمالي التكلفة لكل مدينة.
6. **إنشاء مخطط شلالي:**
   - استخدام `plt.subplots()` لإنشاء Figure و Axes.
   - إنشاء متغير `bottom` لتتبع موضع البدء لشريطات التراكم.
   - استخدام `for i, row in city_totals.iterrows()`:
     - أنشأنا شريطا لكل مدينة باستخدام `ax.bar()`.
     - حددنا موضع البدء (`bottom`) والتسمية (`f"{row['City']} ({row['Percentage']:.2f}%)"`).
     - تحديث `bottom` للمدينة التالية.
   - نعيّن تسميات المحاور (`ax.set_xlabel()`، `ax.set_ylabel()`) وعنوان المخطط (`ax.set_title()`).
   - نضيف تعليقا (`ax.legend()`) لتحديد الشرائح بأسماء المدن والنسب المئوية.
   - تدوير تسميات المحور x (`plt.xticks(rotation=45)`) لتحسين القراءة.
7. **عرض المخطط:**
   - استخدام `plt.show()` لعرض المخطط المُنشأ.

مخرجات الرماز البرمجي الخاص بالمخطط الشلالي:

من هذا المخطط البياني يمكننا استنتاج ما يلي بشكل مختصر:

1 .التوزيع حسب المدن: المخطط يُظهر إجمالي التكلفة لكل مدينة من المدن الموجودة في البيانات.

2 .الترتيب تنازلياً: المدن مرتبة تنازلياً حسب إجمالي التكلفة، مما يساعد في تحديد المدن الأكثر تكلفة.

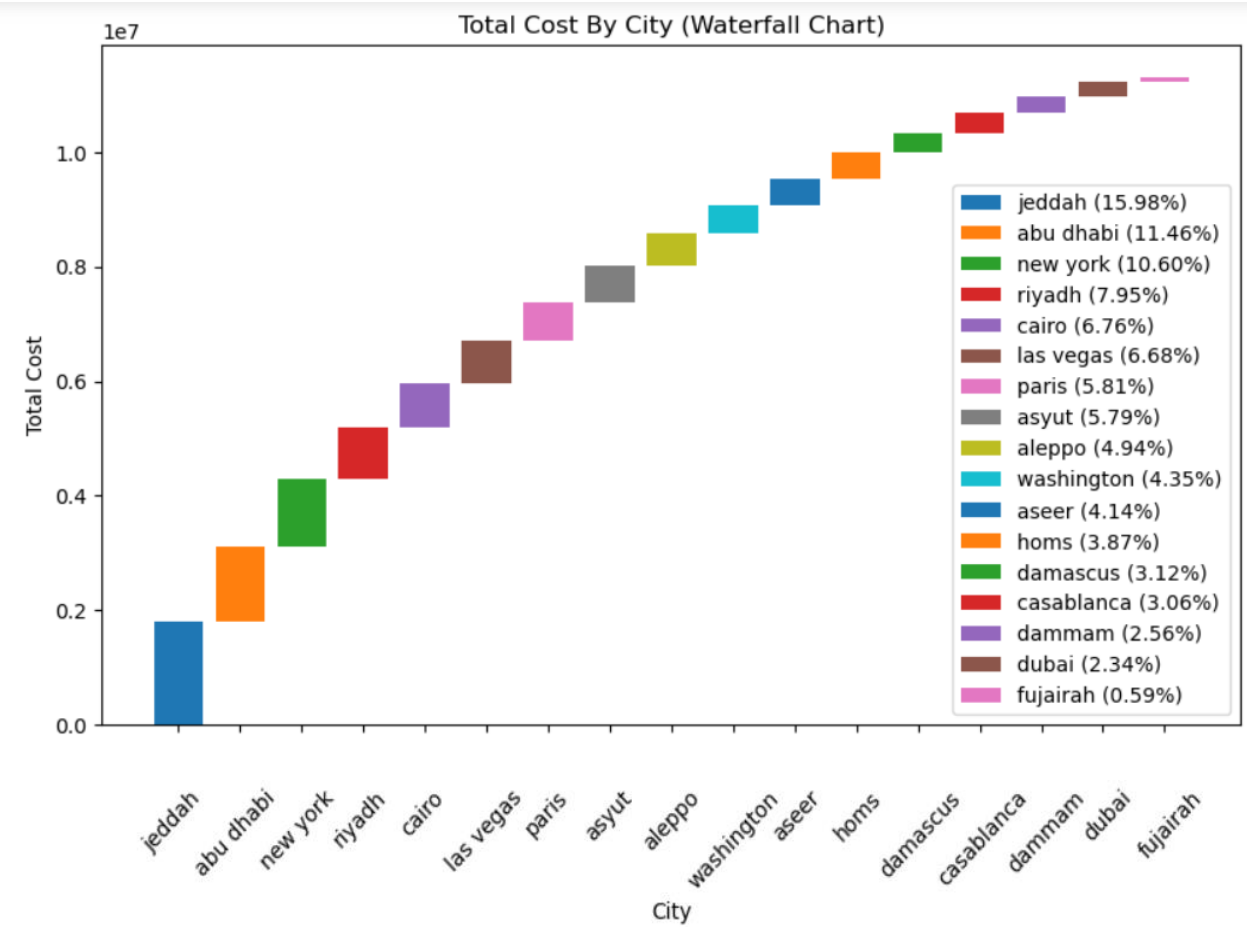
3 .النسب المئوية: المخطط يُظهر النسبة المئوية من إجمالي التكلفة لكل مدينة، وهذا يساعد في فهم مساهمة كل مدينة في إجمالي التكاليف.

4 .الرؤية الشاملة: المخطط النسيابي يعطي صورة شاملة للتكاليف عبر جميع المدن، مما يساعد في التخطيط والتحليل على مستوى المؤسسة.

5 .المقارنة بين المدن: يمكن مقارنة المدن ببعضها البعض من حيث إجمالي التكاليف ونسبها المئوية.

6 .دعم اتخاذ القرار: هذه المعلومات يمكن استخدامها في اتخاذ قرارات بشأن تخصيص الموارد والاستثمارات بين المدن المختلفة.

7 .البساطة والوضوح: استخدام المخطط الانسيابي يجعل العرض بسيطًا وسهل الفهم، خاصةً عند وجود عدد كبير من المدن.



- المخطط الفقاعي Bubble Chart:

الفكرة من هذا المخطط هي تحليل مجموع الخصومات المقدمة على المنتجات لكل فئة وعرضها على شكل مخطط الفقاعات (Bubble Chart) باستخدام مكتبة plotly.express.

```
# Group data by Category and calculate Discount
category_revenue = df.groupby('Category')['Discount'].sum()
# Create the bubble chart using Plotly Express
fig = px.scatter(
    category_revenue,
    x=category_revenue.index,
    y=category_revenue.values,
    size=category_revenue.values,
    color=category_revenue.index,
    hover_name=category_revenue.index,
    size_max=40,
    title="Most Quantity by Country",
)
# Update chart layout with informative axis labels and center the title
fig.update_layout(
    title="Highest Discounts Offered For Each Category",
    title_x=0.5  # Center the title horizontally (0 to 1)
)
# Update chart layout with informative axis labels
fig.update_xaxes(title="Category")
fig.update_yaxes(title="Discount")
# Display the chart
fig.show()
```

أهم الباراميترات في الرماز البرمجي Bubble Chart:

1. **استيراد المكتبات:**
   - `pandas`: لمعالجة البيانات وتحليلها.
   - `plotly. express`: لإنشاء رسومات تفاعلية باستخدام مكتبة Plotly.

- **تجميع البيانات حسب الفئة:**

  - `category_revenue = df.groupby('Category')['Discount'].sum()`:
    - يُجمّع البيانات في إطار البيانات `df` حسب عمود "الفئة".
    - يحسب إجمالي قيمة "الخصم" لكل فئة من المنتجات باستخدام `sum()`.
    - يخزن النتيجة في متغير جديد `category_revenue`.

•إنشاء **مخطط الفقاعات:**

- `fig = px.scatter(...)`:
  - يستخدم `px.scatter` من مكتبة Plotly Express لإنشاء مخطط تشتت.(scatter plot)
  - يستخدم متغير `category_revenue` الذي يحتوي على إجمالي الخصومات لكل فئة.
- `x=category_revenue.index`:
  - يُعيّن قيم المحور الأفقي (x) بأسماء الفئات فهرس .(`category_revenue`)
- `y=category_revenue.values`:
  - يُعيّن قيم المحور العمودي (y) بقيم إجمالي الخصومات.
- `size=category_revenue. values`:
  - يُحدد حجم الفقاعات بناءً على قيم إجمالي الخصومات (أكبر خصم = فقاعة أكبر).
- `color=category_revenue. index`:
  - يُحدد لون الفقاعات بناءً على أسماء الفئات.
- `hover_name=category_revenue. index`:
  - يُظهر اسم الفئة عند تحريك المؤشر فوق الفقاعة.
- `size_max=40`:
  - يُحدد الحد الأقصى لحجم الفقاعات (لمنع التشوش).
- #: `title="Most Quantity by Country"` يحتوي على خطأ
  - يُعدل عنوان المخطط (خطأ في الكود الأصلي، نقوم بتصحيحه لاحقا).

•تخصيص **تخطيط المخطط:(Layout)**

- `fig.update_layout(...)`:
  - يُعدل تخطيط المخطط باستخدام .`update_layout`
- `title="Highest Discounts Offered For Each Category"`:
  - يُعدل عنوان المخطط إلى "أعلى الخصومات المقدمة لكل فئة."
- `title_x=0.5`:
  - يُحّث عنوان المخطط إلى المنتصف أفقيا.
- `fig.update_xaxes(title="Category")`:
  - يُضيف تسمية "فئة" إلى المحور الأفقي.(x)
- `fig.update_yaxes(title="Discount")`:
  - يُضيف تسمية "خصم" إلى المحور العمودي.(y)

•عرض المخطط:

- `fig.show()`:
  - يعرض مخطط الفقاعات المُنشأ.

مخرجات الرماز البرمجي الخاص بمخطط الفقاعي:

من هذا المخطط البياني يمكننا استنتاج ما يلي بشكل مختصر:

1 .تم تجميع البيانات حسب الفئة (Category) وحساب إجمالي الخصم (Discount) لكل فئة.

2 .تم إنشاء مخطط فقاعي (bubble chart) لتمثيل البيانات، حيث تمثل أقطار الفقاعات حجم الخصم لكل فئة.

3 .عنوان المخطط هو "Highest Discounts Offered For Each Category" مما يشير إلى أن المخطط يوضح الخصومات المقدمة الأعلى لكل فئة.

4 .المحور السيني (x-axis) يمثل الفئات(Categories) ، والمحور الصادي (y-axis) يمثل إجمالي الخصم (Discount)لكل فئة.

5. يوضح المخطط البياني أنماط وتباينات في الخصومات المقدمة لكل فئة، مما يساعد في تحليل وفهم سياسات التسعير والترويج لمنتجات كل فئة.

- مخطط ساينكي Sankey:

الفكرة الأساسية من هذا المخطط هي تحليل مجموع الكميات المباعة حسب الجنس والفئة ويعرضها على شكل مخطط Sankey تفاعلي باستخدام مكتبة holoviews.

```
import holoviews as hv# Import the Holoviews library for creating interactive visualizations
# Group data by Gender and Category, sum Quantity
data = df.groupby(['Gender','Category'])['Quantity'].sum().reset_index()

# Enable HTML output for visualization
%env HV_Doc_HTML=True
hv.extension('bokeh')

# Create the Sankey plot
sankey = hv.Sankey(
    data,
    kdims=['Gender','Category'],label="Top Selling Categories among Males and Females "  # Clearer label
)
# Customize visualization options
sankey = sankey.opts(
    label_position='left', # Place labels on the left
    edge_color='Gender',    # Color edges based on Region
    node_color='index',     # Color nodes based on index (category order)
)
# Display the Sankey plot
sankey
```

أهم الباراميترات في الرماز البرمجي الخاص بمخطط (Sankey):

1. **استيراد المكتبات:**
   - `pandas`: لمعالجة البيانات وتحليلها.
   - `holoviews`: لإنشاء رسومات تفاعلية.
2. **تجميع البيانات:**

- تجميع البيانات في إطار البيانات `df` حسب كل من "الجنس" و "الفئة" لحساب إجمالي "الكمية" المباعة.
- إعادة ضبط الفهرس لإنشاء إطار بيانات جديد `data` يحتوي على البيانات المجمعة.

3. **تمكين إخراج HTML وإنشاء مخطط Sankey:**

- تمكين إخراج HTML لعرض المخطط في متصفح الويب.
- استخدام `hv.Sankey` لإنشاء مخطط Sankey ، ومرّر البيانات المُجمعة `data` كحجة.
- تحديد الأبعاد الرئيسية (`kdims`) كـ "الجنس" و "الفئة" لتمثيل اتجاهات التدفق في المخطط.

- إضافة ملصق وصفي للمخطط باستخدام `label`.

4 . **تخصيص مخطط:Sankey**

- تعيين `label_position='left'` لوضع العلامات على الجانب الأيسر من المخطط.
- تعيين `edge_color='Gender'` لتلوين الحواف بناءً على "الجنس" (ذكر أو أنثى).
- تعيين `node_color='index'` لتلوين العقد بناءً على الفهرس (ترتيب فئات المنتجات).

5. **عرض المخطط:**

- عرض مخطط Sankey باستخدام `sankey`.

مخرجات الرماز البرمجي الخاص بمخطط Sankey:

من خلال هذا المخطط البياني يمكن استنتاج ما يلي بشكل مختصر:

1 .توزيع المبيعات حسب الجنس: المخطط السانكي يُظهر توزيع المبيعات لكل فئة منتجات بين الذكور والإناث. هذا يساعد في فهم أنماط الاستهلاك والتفضيلات المختلفة بين الجنسين.

2 .الفئات الأكثر مبيعاً لكل جنس: من خلال المخطط، يمكننا تحديد الفئات المنتجات الأكثر مبيعاً بالنسبة للذكور والإناث على حدٍ سواء. هذه المعلومات قد تكون مفيدة في التسويق والاستهداف الموجه.

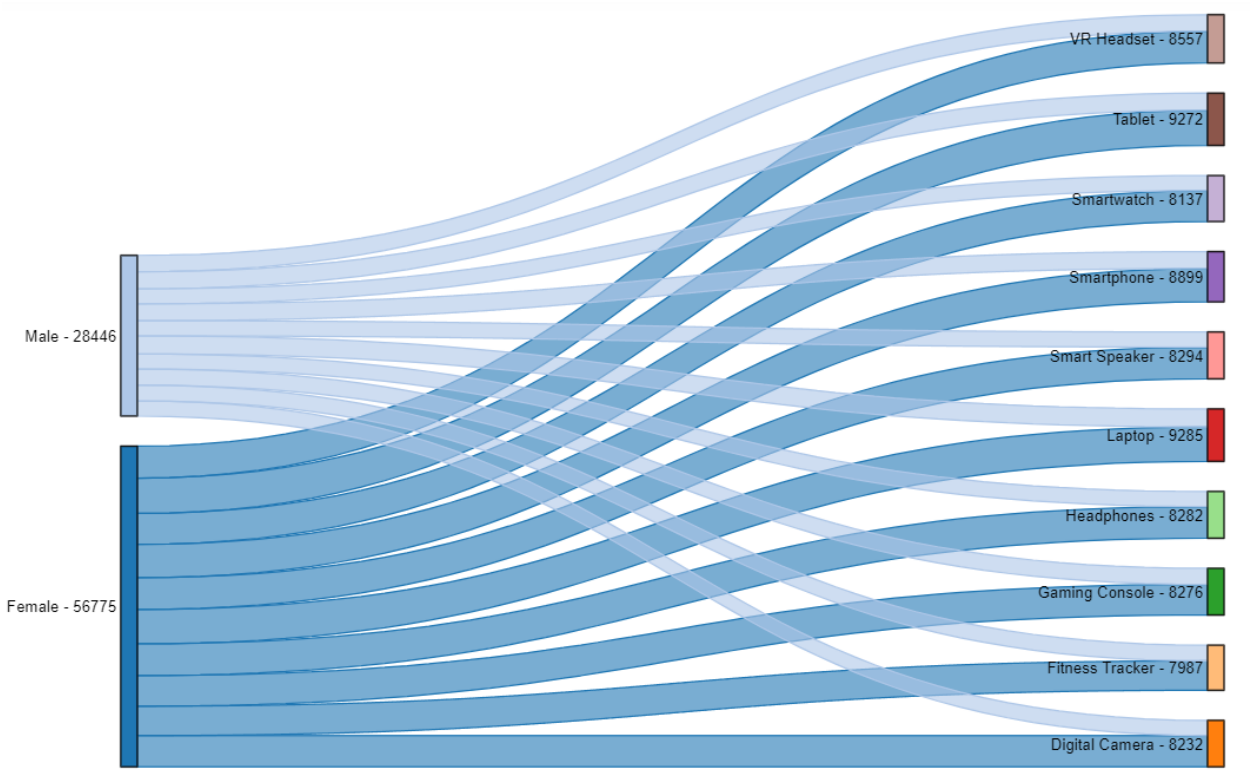
3 .الاختلافات في الاستهلاك: المخطط يُظهر الاختلافات في حجم المبيعات لكل فئة منتجات بين الذكور والإناث. هذه المعلومات يمكن استخدامها في تطوير استراتيجيات تسويقية مخصصة لكل جنس.

4 .تحديد الفرص: من خلال تحليل الاختلافات في المبيعات بين الجنسين، يمكن تحديد الفرص المحتملة لزيادة المبيعات في الفئات التي يكون لها فجوة كبيرة بين الذكور والإناث.

5 .الصورة الكلية: المخطط السانكي يوفر لمحة عامة شاملة عن توزيع المبيعات بين الجنسين لجميع فئات المنتجات، مما يساعد في استراتيجيات التسويق والتخطيط الشامل.

6 .سهولة التفسير: تقديم البيانات في شكل مخطط سانكي يجعل من السهل فهم العلاقات والاتجاهات بين الجنس وفئات المنتجات بشكل سريع ومباشر.

```
env: HV_Doc_HTML=True
```



- **مخطط أشعة الشمس Sunbrust:**

الفكرة الأساسية من هذا المخطط هي تحليل بيانات المبيعات لعرض أعلى المنتجات مبيعاً في كل مدينة داخل كل دولة(Country) باستخدام مكتبة plotly.express.

```python
# Group the data by 'Country' and ' City', and sum the 'Quantity'
data = pd.DataFrame(df.groupby(['Country', 'City']).agg({'Quantity': 'sum'}))

# Create a sunburst chart to visualize the quantity based on country and city
fig = px.sunburst(df, path=['Country', 'City'], values='Quantity')

# Update the layout of the chart
fig.update_layout(title_text="Best Selling Quantities In Each City Within The Country", title_x=0.5)

# Display the chart
fig.show()
```

أهم الباراميترات في الرماز البرمجي الخاص بمخطط أشعة الشمس Sunbrust:

1. **استيراد المكتبات:**
   - `pandas`: لمعالجة البيانات وتحليلها.
   - `plotly. express`: لإنشاء رسومات تفاعلية باستخدام مكتبة Plotly.
2. **تجميع البيانات وحساب إجمالي المبيعات:**
   - استخدام `df.groupby(['Country', 'City']).agg({'Quantity': 'sum'})`

لإنشاء DataFrame جديد `data` يُظهر إجمالي "الكمية المباعة" لكل مدينة داخل كل بلد.

- `groupby(['Country', 'City'])`: يُجمع البيانات حسب عمودي "البلد" و "المدينة".
- `agg({'Quantity': 'sum'})`: يُلخص "الكمية المباعة" لكل مجموعة بحساب المجموع.

3. **إنشاء مخطط Sunburst:**
   - استخدام `px.sunburst(data,path=['Country', 'City'], values='Quantity')` لإنشاء مخطط Sunburst باستخدام `plotly.express`.
     - `data`: DataFrame يحتوي على إجمالي "الكمية المباعة" لكل مدينة داخل كل بلد.
     - `path=['Country', 'City']`: يُحدد هرمية المخطط.
       - المستوى الخارجي: يُمثل البلدان.
       - المستوى الداخلي: يُمثل المدن داخل كل بلد.
     - `values='Quantity'`: يُحدد "الكمية المباعة" لتحديد حجم القطاعات في المخطط.

4. **تحديث تخطيط المخطط:**
   - استخدام `fig.update_layout(title_text="Best Selling Quantities In Each City Within The Country", title_x=0.5)` لتغيير تخطيط المخطط.
     - `title_text="Best Selling Quantities In Each City Within The Country"`: يُحدد عنوان المخطط.
     - `title_x=0.5`: يُمركز عنوان المخطط أفقيا.
5. **عرض المخطط:**
   - استخدم `fig.show()`: لعرض المخطط المُنشأ.

مخرجات الرماز البرمجي الخاص بالمخطط أشعة الشمس التفاعلي Sunbrust:

من خلال هذا المخطط البياني يمكننا استنتاج ما يلي بشكل مختصر:

1 .تم تجميع البيانات حسب البلد (Country) والمدينة(City) ، وحساب إجمالي الكمية (Quantity) لكل منهما.

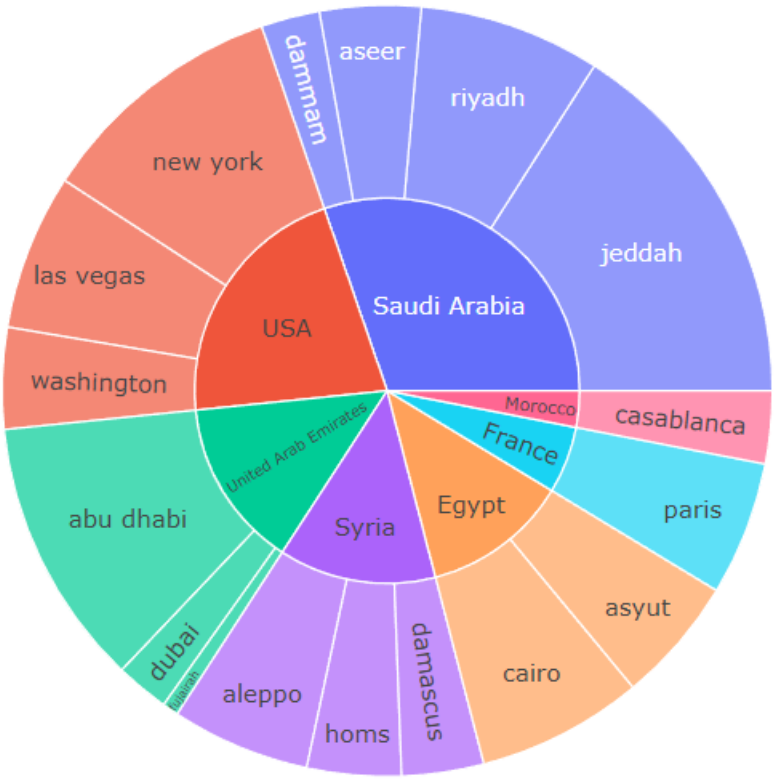
2 .المخطط يوضح أفضل الكميات المباعة في كل مدينة ضمن كل بلد.

3 .يتم عرض البيانات في شكل هرمي، حيث يمثل المستوى الأعلى البلد والمستوى الأدنى يمثل المدينة.

4 .حجم شرائح المخطط يعكس كمية المبيعات لكل مدينة داخل البلد، مما يسمح بمقارنة الأداء بين المدن والبلدان.

5. هذا المخطط يساعد في تحديد أفضل المناطق والمدن من حيث حجم المبيعات، مما قد يساعد في وضع استراتيجيات تسويقية وتوزيعية أكثر استهدافاً.

Best Selling Quantities In Each City Within The Country



- الخريطة الشجرية Treemap:

الفكرة الأساسية من هذا المخطط هي تحليل مجموع الكميات المباعة حسب كل فئة فرعية

(Sub Category) ضمن كل فئة رئيسية (Category)، ويعرضها على شكل مخطط Treemap (خريطة شجرية) باستخدام مكتبة plotly. express.

```
# Group the data by Category and Sub Category, and sum the Quantity
data = pd.DataFrame(df.groupby(['Category', 'Sub Category']).agg({'Quantity':'sum'}))

# Create a treemap plot to visualize the Quantity by Category and Sub Category
fig = px.treemap(df,
                 path=['Category','Sub Category'],
                 values='Quantity',
                 color='Category',
                 color_discrete_map={'(?)': 'lightgrey'})

# Update the root node color to black
fig.update_traces(root_color="black")

# Set the title and center it
fig.update_layout(title_text="Top Selling Quantities Sub Category In Each Category", title_x=0.5)

# Display the treemap plot
fig.show()
```

أهم الباراميترات في الرماز البرمجي الخاص بمخطط Treemap:

1. **استيراد المكتبات :**
   - `pandas`: لمعالجة البيانات وتحليلها.
   - `plotly. express`: لإنشاء رسومات تفاعلية باستخدام مكتبة Plotly.
2. **تجميع البيانات وحساب إجمالي المبيعات:**
   - استخدام `df.groupby(['Category', 'Sub Category']).agg({'Quantity': 'sum'})` لإنشاء DataFrame جديد `data` يُظهر إجمالي "الكمية المباعة" لكل فئة فرعية داخل كل فئة.
     - `groupby(['Category', 'Sub Category'])`:

       يُجمع البيانات حسب عمودي "الفئة" و "الفئة الفرعية."
     - `agg({'Quantity': 'sum'})`:

       يلخص "الكمية المباعة" لكل مجموعة بحساب المجموع.

3. **إنشاء الخريطة الشجرية:**

- استخدم `px.treemap(data, path=['Category', 'Sub Category'], values='Quantity', color='Category', color_discrete_map={'(?)':'lightgrey'})` لإنشاء مخطط شجرة باستخدام plotly.express.
  - `data`: DataFrame يحتوي على إجمالي "الكمية المباعة" لكل فئة فرعية داخل كل فئة.
  - `path=['Category', 'Sub Category']`: يُحدد هرمية المخطط.
    - ✓ المستوى العلوي: يُمثل الفئات.
    - ✓ المستوى السفلي: يُمثل الفئات الفرعية داخل كل فئة.
  - `values='Quantity'`: يُحدد "الكمية المباعة" لتحديد حجم المستطيلات في المخطط.
  - `color='Category'`: يُستخدم "الفئة" لتلوين المستطيلات.
  - `color_discrete_map={'(?)': 'lightgrey'}`: يُحدد لوناً افتراضياً (رمادي فاتح) للفئات ذات القيم المفقودة الممثلة بـ "(?)" في عمود "الفئة."

4. **تخصيص لون العقدة الجذرية (اختياري):**

- `fig.update_traces(root_color="black")`:

يُغير لون المستطيل العلوي (يمثل جميع الفئات) إلى الأسود.

5. **تحديث تخطيط المخطط:**

- `fig.update_layout(title_text="Top Selling Quantities Sub Category In Each Category", title_x=0.5)`:

يُحدد عنوان المخطط إلى "Top Selling Quantities Sub Category In Each Category" ويُمركز العنوان أفقياً.

6. **عرض المخطط:**

- `fig.show()`: لعرض المخطط المُنشأ.

مخرجات الرماز البرمجي الخاص بمخطط Treemap:

من هذا المخطط البياني يمكننا استنتاج ما يلي بشكل مختصر:

1 .تم تجميع البيانات حسب الفئة والفئة الفرعية، وتم حساب المجموع الكلي للكمية لكل فئة فرعية.

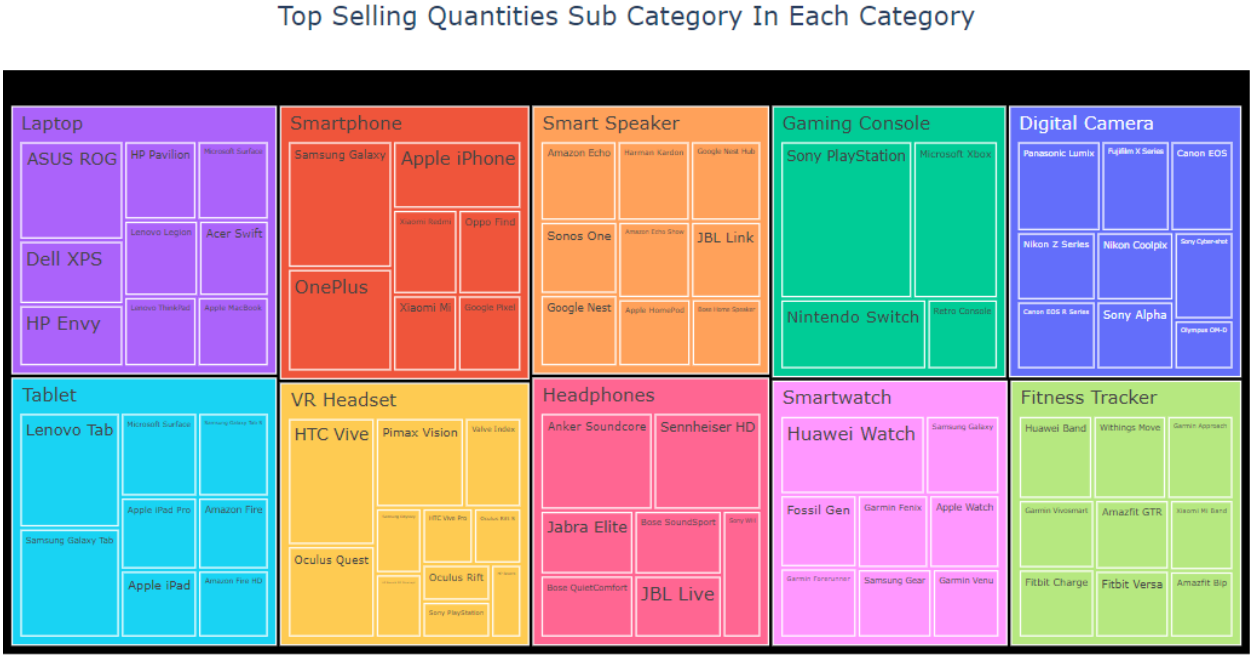
2 .تم إنشاء مخطط شجري (treemap) لتمثيل الكميات حسب الفئة والفئة الفرعية.

3 .لون العقدة الجذرية تم تحديده باللون الأسود.

4 .تم إضافة عنوان مركزي للمخطط

"Top Selling Quantities Sub Category In Each Category" لتوضيح محتوى المخطط.

5. يوضح المخطط البياني الأنماط والاتجاهات في الكميات المباعة لكل فئة فرعية داخل كل فئة رئيسية.



- الخريطة الحرارية Heatmap:

الفكرة الأساسية من هذا المخطط هي تحليل كميات المبيعات لكل فئة حسب الشهر ويعرضها على شكل خريطة حرارية (Heatmap) باستخدام مكتبةseaborn .

```
# Extract the month from the 'Order Date' column and create a new 'Month' column
df['Month'] = pd.to_datetime(df['Order Date'], format='mixed').dt.month_name().sort_values()
# Restructure data to have Month as index and Sales for each category
data = df.pivot_table(index="Month", columns="Category", values="Quantity")
# Reorder the index to sort the months from smallest to largest
months = ['January', 'February', 'March', 'April', 'May', 'June',
          'July', 'August', 'September', 'October', 'November', 'December']
data = data.reindex(sorted(data.index, key=lambda x: months.index(x)))
# Create the heatmap using Seaborn
fig, ax = plt.subplots(figsize=(12, 8))
cmap = "RdBu_r"
sns.heatmap(data, ax=ax, cmap=cmap, vmin=data.min().min(), vmax=data.max().max(),
            cbar_kws={"shrink": 0.5},
            cbar_ax=ax.figure.add_axes([0.92, 0.15, 0.02, 0.7]))
# Get category labels from the heatmap
category_labels = list(data.columns)
# Set x-axis ticks and labels with rotation using Matplotlib
ax.set_xticks(np.arange(len(category_labels)))  # Set tick positions
ax.set_xticklabels(category_labels, rotation=45, ha="right")  # Set labels with rotation
# Add a colorbar with a title
cbar = ax.figure.colorbar(sns.heatmap(data, ax=ax, cmap=cmap, vmin=data.min().min(),
                                      vmax=data.max().max(), cbar_kws={"shrink": 0.5},
                                      cbar_ax=ax.figure.add_axes([0.92, 0.15, 0.02, 0.7])).collections[0],
                          cax=ax.figure.add_axes([0.92, 0.15, 0.02, 0.7]))
cbar.ax.set_ylabel('Quantity', rotation=-90, va="bottom")
# Add labels and title
ax.set_xlabel("Category")
ax.set_ylabel("Month")
ax.set_title("Top Selling Categories By Month")
# Show the heatmap
plt.show()
```

أهم الباراميترات في الرماز البرمجي الخاص بمخطط Heatmap:

2. **استيراد المكتبات:**
   - `pandas`: لمعالجة البيانات وتحليلها.
   - `seaborn`: لإنشاء رسومات إحصائية جذابة وواضحة.
   - `matplotlib.pyplot`: لإنشاء عناصر الرسم البياني الأساسية.

- استخراج **الشهر وإنشاء عمود جديد:**

- `df['Month']=pd.to_datetime(df['Order Date'], format='mixed').dt.month_name().sort_values()`:
  - يستخرج الشهر من عمود "تاريخ الطلب".

- يحوله إلى اسم الشهر (يناير، فبراير، إلخ).
- يُنشئ عموداً جديداً باسم "Month" ويُضيف أسماء الشهور بعد فرزها تصاعدياً.

	Order ID	Order Date	Day	Country	City	Lat	Lng	Full Name	Category	Sub Category	Item	SalesPerson ID	Quantity	Unit Price	Discount	Total Cost	Status	Gender	Month
0	1	01/01/2023	Mon	Syria	homs	34.7326	36.7136	Lina Alrrashid	Tablet	Apple iPad	iPad Pro 12.9"	N498	4	999	38.3616	891.9072	False	Female	January
1	2	01/01/2023	Tue	Saudi Arabia	riyadh	24.7136	46.6753	Omar Eurul	Smartphone	Samsung Galaxy	Galaxy S21 Ultra	X918	3	1199	517.9680	302.1480	True	Female	January
2	3	01/01/2023	Tue	Saudi Arabia	riyadh	24.7743	46.7386	Iman Ismaeil	Digital Camera	Panasonic Lumix	Panasonic Lumix GH5	I036	4	1299	883.3200	831.3600	True	Male	January
3	4	01/01/2023	Mon	United Arab Emirates	abu dhabi	24.4539	54.3773	Ahmad Rihan	Tablet	Samsung Galaxy Tab	Galaxy Tab A8	E804	6	199	33.3126	129.5490	True	Female	January
4	5	01/01/2023	Wed	USA	washington	38.9072	-77.0369	Sami Altawil	Headphones	Sennheiser HD	Sennheiser HD 450BT	Q149	4	129	11.2617	111.3657	True	Female	January
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
19995	19996	01/09/2023	Tue	Morocco	casablanca	33.5731	7.5898	Ahmad Iad	Gaming Console	Sony PlayStation	PlayStation 4 Pro	M210	4	399	150.9417	178.0338	True	Female	January
19996	19997	01/09/2023	Tue	Syria	homs	34.7326	36.7136	Ali Kiali	Smartwatch	Fossil Gen	Fossil Gen 6	Z826	3	299	398.2680	215.2800	True	Female	January
19997	19998	01/09/2023	Wed	USA	las vegas	36.1699	-115.1398	Husayn Salayk	Gaming Console	Sony PlayStation	PlayStation 2	K624	5	99	15.3648	56.6577	True	Male	January
19998	19999	01/09/2023	Mon	Saudi Arabia	jeddah	21.4858	39.1925	Fatin Bahriin	Headphones	JBL Live	JBL Live 500BT	Y368	4	99	19.9584	68.4288	True	Female	January
19999	20000	01/09/2023	Tue	USA	washington	38.9072	-77.0369	Ahmad Shakur	Smartwatch	Samsung Galaxy	Galaxy Watch Active 2	Y590	4	249	19.5216	180.5748	True	Female	January

- إعادة **هيكلة البيانات:**

- `data=df.pivot_table(index="Month",columns="Category", values="Quantity")`:
  - يعيد هيكلة البيانات باستخدام `pivot_table` من مكتبة Pandas.
  - يجعل "Month" هو المُؤشر (الصفوف).
  - يجعل "Category" الأعمدة.
  - يضع قيم "Quantity" في كل خلية بناءً على الشهر والفئة.

- إعادة **ترتيب الشهور:**

- `months = [...]`:
  - يُنشئ قائمة تحتوي على أسماء الشهور بالترتيب.
- `data = data.reindex(sorted(data.index, key=lambda x: months.index(x)))`:
  - يعيد ترتيب الشهور في المُؤشر (الصفوف) وفقاً للقائمة التي تم إنشاؤها.

●إنشاء **المخطط الحراري:**

- `fig, ax = plt.subplots(figsize=(12, 8))`:
  - يُنشئ مجسم (Figure) ومحور (Axes) للمخطط.
- `cmap = "RdBu_r"`:
  - يُحدد تدرج الألوان للمخطط الحراري(RdBu_r) مناسب للبيانات التي تتراوح من سالبة إلى موجبة.
- `sns.heatmap(data, ax=ax, cmap=cmap, ...)`:
  - يستخدم مكتبة Seaborn لإنشاء المخطط الحراري.
  - يُغذى المخطط بالبيانات المُعاد ترتيبها `data`.
  - يُحدد تدرج الألوان المُختار ومدى القيم (الحد الأدنى والحد الأقصى) للمخطط.
- `cbar = [...]`:
  - يُنشئ شريط الألوان (Colorbar) الذي يُظهر تدرج الألوان وقيمها.
- `ax.set_xticks(...), ax.set_xticklabels(...)`:
  - يُحدد المواضع والتعليقات على محور) x الذي يمثل الفئات.(
  - تُدوير تعليقات الفئات بزاوية 45 درجة لتحسين القراءة.
- `cbar.ax.set_ylabel('Quantity', ...)`:
  - يُضيف تسمية لمحور شريط الألوان (الكمية).
- `ax.set_xlabel(...), ax.set_ylabel(...), ax.set_title(...)`:
  - يُضيف تسميات للمحاور (الفئة، الشهر) والعنوان للمخطط (أفضل المنتجات مبيعاً حسب الشهر).

●**عرض المخطط:**

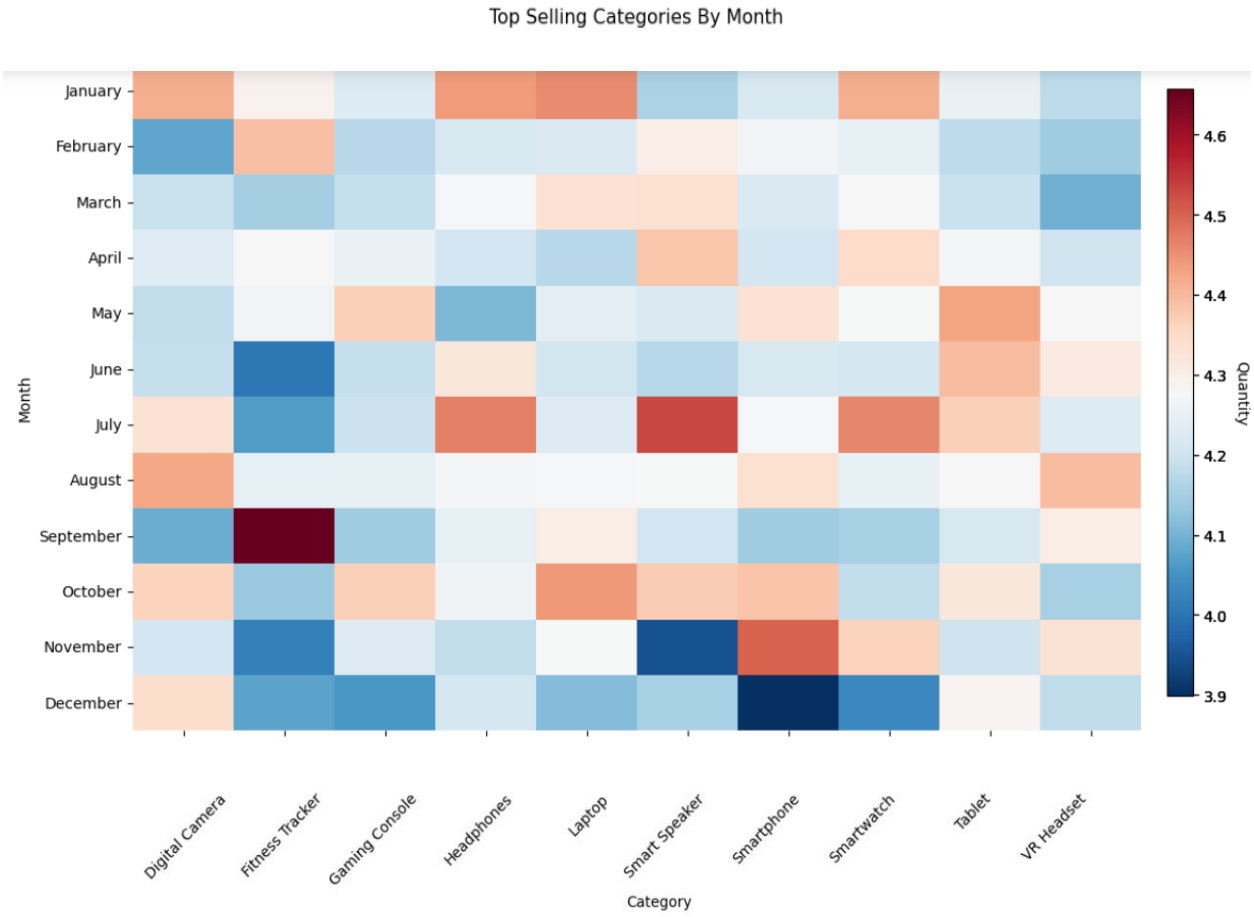
- `plt.show()`:
  - يعرض المخطط الحراري المُنشأ.

مخرجات الرماز البرمجي الخاص بمخطط Heatmap:

من هذا المخطط البياني يمكننا استنتاج ما يلي بشكل مختصر:

1. التغير الشهري في المبيعات: المخطط الحراري يُظهر التغيرات في المبيعات لكل فئة منتجات على مدار الأشهر. هذا يساعد في تحديد الأشهر ذات المبيعات العالية والمنخفضة لكل فئة.

2. الفئات الأكثر مبيعاً: من خلال المخطط، يمكننا تحديد الفئات الأكثر مبيعاً في كل شهر، وهذا قد يكون مفيداً في التخطيط والتسويق.
3. الاتجاهات الموسمية: المخطط يُظهر الاتجاهات الموسمية في المبيعات لكل فئة منتجات، مما يساعد في التخطيط الاستراتيجي والتنبؤ بالطلب.
4. تحديد الفرص: من خلال تحليل أنماط المبيعات الشهرية، يمكن تحديد الفرص المحتملة لزيادة المبيعات في الأشهر ذات المبيعات المنخفضة.
5. تحليل الأداء: المخطط الحراري يوفر لمحة عامة عن أداء كل فئة منتجات على مدار السنة، مما يساعد في تقييم الأداء والاتخاذ القرارات المناسبة.
6. سهولة التفسير: تقديم البيانات في شكل مخطط حراري يجعل من السهل فهم الاتجاهات والنمط العام للمبيعات بشكل سريع ومباشر.



خاتمة: رحلة عبر عالم التمثيل البصري

لقد قمنا بجولة غنية عبر تقنيات التمثيل البصري المتنوعة، وشاهدنا كيف يمكن تحويل البيانات المجردة إلى رسومات وصور تخاطب عقولنا وتجذب انتباهنا. من المخططات الشريطية العمودية والأفقية إلى مخططات ساينكي والشلالية، ومن الخرائط الشجرية والحرارية إلى أشعة الشمس والمخططات الدائرية والشريطية المقارنة، برزت قدرة هذه الأدوات على تبسيط المعقد، وتوضيح الغامض، وكشف العلاقات المخفية.

ولكن رحلتنا لا تنتهي هنا، فما زال عالم التمثيل البصري يتطور باستمرار، وتظهر تقنيات جديدة كل يوم. وعلينا أن نبقى على اطلاع دائم بهذه التطورات، وأن نختار الأدوات المناسبة لاحتياجاتنا، ونستخدمها بذكاء وإبداع لتوصيل رسالتنا بشكل فعّال.

في الختام،

- التّمثيل البصري لغة عالمية تتخطى اللغات والثقافات.
- هو أداة قوية لفهم العالم من حولنا واتخاذ القرارات.
- علينا أن نتقن استخدام هذه اللغة ونوظفها لخدمة أهدافنا.

ولكن يجب أن نتذكر:

- لا توجد تقنية واحدة مثالية لكل شيء.
- اختيار التمثيل البصري المناسب للبيانات.
- استخدام الألوان والتصميمات بشكل إبداعي لجذب الانتباه.
- تبسيط الرسومات وتجنب الزحام.
- التركيز على ما هو مهم وترك البيانات تتحدث عن نفسها.

مع استخدام ذكي وإبداعي، يمكن أن يصبح التمثيل البصري أداة قوية للتواصل والتعلم والإبداع.

المراجع:

- Embarak, D. O., & Embarak, O. (2018). Data visualization. *Data Analysis and Visualization Using Python: Analyze Data to Create Visualizations for BI Systems*, 293-342.
- Sial, A. H., Rashdi, S. Y. S., & Khan, A. H. (2021). Comparative analysis of data visualization libraries Matplotlib and Seaborn in Python. *International Journal*, *10*(1), 45.
- Petrou, T. (2017). *Pandas Cookbook: Recipes for Scientific Computing, Time Series Analysis and Data Visualization using Python*. Packt Publishing Ltd.
- Waskom, M. L. (2021). Seaborn: statistical data visualization. Journal of Open Source Software, 6(60), 3021.
- Lavanya, A., Sindhuja, S., Gaurav, L., & Ali, W. (2023). A comprehensive review of data visualization tools: features, strengths, and weaknesses. Int. J. Comput. Eng. Res. Trends, 10(01), 10-20.
- Abdelalim, A., & O'Brien, W. (2017, May). Visualization of building performance using sankey diagrams to enhance the decision-making process. In *Proceedings of the Symposium on Simulation for Architecture and Urban Design* (pp. 1-8).
- Stehlík, B. V. Developing Effective Big Data Analytics by Leveraging Visualization Techniques.