CMPE 256 Recommender Systems Data Science Hackathon

❖ Electronic Retailer Market Basket Recommendation System for Online Checkout Integration

❖ Overview :-

For the first part of the CMPE 256 Data Science Hackathon, my team and I developed an Electronic Retailer Market Basket Recommendation System to improve the online checkout experience for customers. Our goal was to build a model that could intelligently recommend products that are frequently bought together based on historical transaction data. We started by cleaning and preparing the dataset, converting transaction records into a one-hot encoded format suitable for analysis. We then applied the Apriori algorithm and association rule mining techniques to discover meaningful relationships between products.
Once the model generated strong association rules, we integrated it into an interactive interface using Python, pandas, mlxtend, and Gradio. This allowed us to simulate a real-time shopping cart where users could add products and instantly receive relevant recommendations. We also ensured that the system dynamically updated suggestions every time a new item was added to the cart. Our team focused on making the recommender both accurate and user-friendly, demonstrating how data-driven insights can enhance e-commerce checkout systems by promoting cross-selling and improving the overall customer experience.

❖ Technical Implementation:-

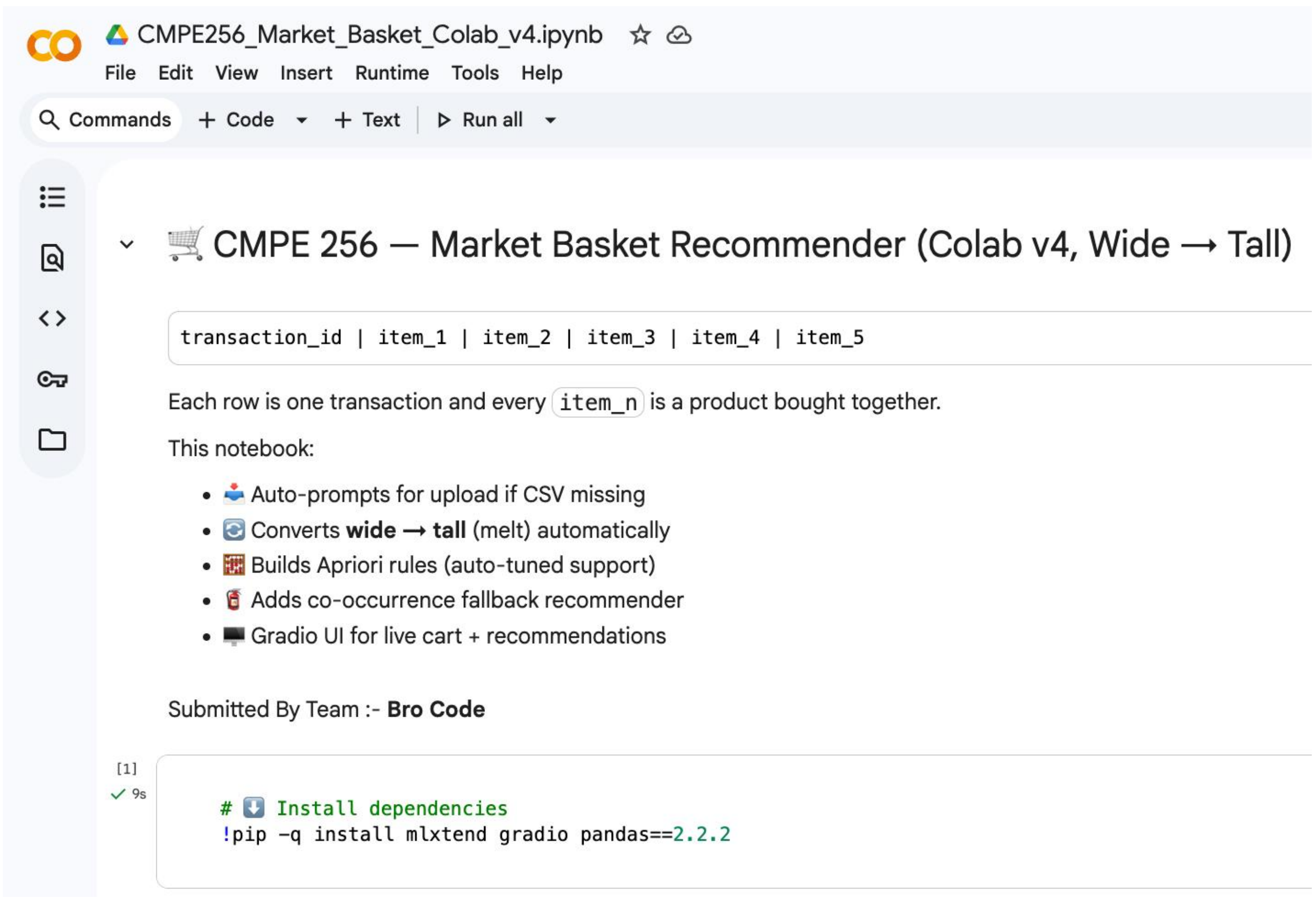
For the first part of the CMPE 256 Data Science Hackathon, my team and I developed an Electronic Retailer Market Basket Recommendation System aimed at improving the online checkout experience through intelligent product recommendations. We began by loading and exploring the provided transaction dataset, which contained product names, SKUs, and transaction IDs. Using Python and libraries like pandas and mlxtend, we performed data preprocessing to clean the dataset and convert it into a one-hot encoded format suitable for association rule mining.

Once the data was prepared, we applied the Apriori algorithm from mlxtend.frequent_patterns to identify frequent itemsets—groups of products that were commonly purchased together. We then used association rule mining to generate meaningful rules based on metrics such as support, confidence, and lift. These rules helped us determine which products should be recommended when a specific item or combination of items was added to a user's shopping cart.

To make the system interactive, we built a simple yet functional Gradio interface. This allowed users to simulate an online checkout experience by selecting products and viewing dynamic recommendations in real time. Every time a new item was added to the virtual cart, the model recalculated the association rules and updated the recommendations accordingly.

Finally, we deployed the recommender in a Google Colab environment, ensuring it was lightweight, reproducible, and easy to demonstrate. Through this implementation, our team successfully created a working prototype that showcased how market basket analysis and association rule learning can enhance cross-selling strategies and customer engagement in an e-commerce setting.

❖ Screenshots :-


CMPE256_Market_Basket_Colab_v4.ipynb
File Edit View Insert Runtime Tools Help
Commands + Code + Text ▷ Run all

🛒 CMPE 256 — Market Basket Recommender (Colab v4, Wide → Tall)

```
transaction_id | item_1 | item_2 | item_3 | item_4 | item_5
```

Each row is one transaction and every `item_n` is a product bought together.

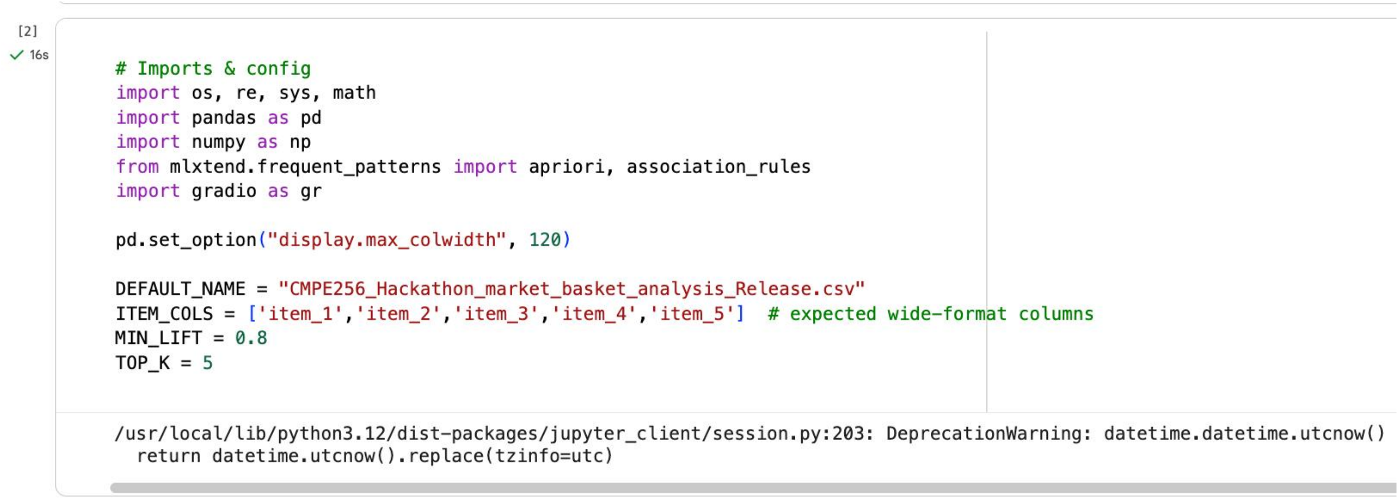
This notebook:

- 📥 Auto-prompts for upload if CSV missing
- 🔄 Converts **wide → tall** (melt) automatically
- 🧮 Builds Apriori rules (auto-tuned support)
- 🧯 Adds co-occurrence fallback recommender
- 🖥️ Gradio UI for live cart + recommendations

Submitted By Team :- **Bro Code**

[1] ✓ 9s

```
# ⬇️ Install dependencies
!pip -q install mlxtend gradio pandas==2.2.2
```



[2] ✓ 16s

```
# Imports & config
import os, re, sys, math
import pandas as pd
import numpy as np
from mlxtend.frequent_patterns import apriori, association_rules
import gradio as gr

pd.set_option("display.max_colwidth", 120)

DEFAULT_NAME = "CMPE256_Hackathon_market_basket_analysis_Release.csv"
ITEM_COLS = ['item_1','item_2','item_3','item_4','item_5']  # expected wide-format columns
MIN_LIFT = 0.8
TOP_K = 5
```

```
/usr/local/lib/python3.12/dist-packages/jupyter_client/session.py:203: DeprecationWarning: datetime.datetime.utcnow()
  return datetime.utcnow().replace(tzinfo=utc)
```

📥 Upload CSV (auto-prompt if missing)

[3]
✓ 18s

```
dataset_path = DEFAULT_NAME
if not os.path.exists(dataset_path):
    try:
        from google.colab import files
        print("File not found:", dataset_path)
        print("Please upload your CSV…")
        uploaded = files.upload()
        if uploaded:
            dataset_path = list(uploaded.keys())[0]
            print("Using uploaded file:", dataset_path)
        else:
            print("No file uploaded; will run with a tiny demo dataset.")
    except Exception as e:
        print("Upload not available (not running in Colab?):", e)
else:
    print("Found dataset:", dataset_path)
```

```
  return datetime.utcnow().replace(tzinfo=utc)
/usr/local/lib/python3.12/dist-packages/jupyter_client/session.py:203: DeprecationWarning: datetime.
  return datetime.utcnow().replace(tzinfo=utc)
/usr/local/lib/python3.12/dist-packages/jupyter_client/session.py:203: DeprecationWarning: datetime.
  return datetime.utcnow().replace(tzinfo=utc)
```

Choose files CMPE256_…Release.csv

CMPE256_Hackathon_market_basket_analysis_Release.csv(text/csv) - 177086 bytes, last modified: 01/11/2025 - 100% done

🧼 Load & preview

Tries multiple encodings. If no CSV, builds a small demo that matches your schema.

[4]
✓ 0s

```
def load_dataframe(path):
    if os.path.exists(path):
        last_err = None
        for enc in ("utf-8", "utf-8-sig", "latin-1"):
            try:
                df = pd.read_csv(path, encoding=enc)
                print(f"Loaded with encoding: {enc}")
                print("Columns:", list(df.columns))
                display(df.head(5))
                return df
            except Exception as e:
                last_err = e
        raise last_err
    else:
        print("⚠️ CSV not found. Using a tiny demo dataset in wide format.")
        demo = pd.DataFrame({
            "transaction_id": ["T1","T2","T3","T4","T5"],
            "item_1": ["Bosch D7050 Detector (SKU: D7050)", "DSC PG9914 Motion Detector (SKU: PG9914)",
                       "Axis M3046-V Network Camera (SKU: 0806-001)", "GE Interlogix 60-652-95R Carbon Monoxide Detector (SKU: 60-652-95R)",
                       "DSC WS4916 Smoke Detector (SKU: WS4916)"],
            "item_2": ["DSC PG9914 Motion Detector (SKU: PG9914)", "Bosch F220-B6 Detector Base (SKU: F220-B6)",
                       "Bosch NVR-5500-16A00 NVR (SKU: NVR-5500-16A00)", "Dahua 8-Channel NVR (SKU: NVR4208-8P-4KS2)",
                       "DSC WS4939 Wireless Key (SKU: WS4939)"],
            "item_3": ["", "", "", "", ""],
            "item_4": ["", "", "", "", ""],
            "item_5": ["", "", "", "", ""],
        })
        display(demo)
        return demo

df_raw = load_dataframe(dataset_path)
```

```
/usr/local/lib/python3.12/dist-packages/jupyter_client/session.py:203: DeprecationWarning: datetime.datetime.utcnow() is deprecated and scheduled for removal in a future version. Use timezone-aware objects
  return datetime.utcnow().replace(tzinfo=utc)
```

	transaction_id	item_1	item_2	item_3	item_4	item_5
0	1	Honeywell 5800CO Carbon Monoxide Detector (SKU: 5800CO)	Hanwha XRN-2010 NVR (SKU: XRN-2010)	NaN	NaN	NaN
1	2	Hanwha QNV-6010R Network Camera (SKU: QNV-6010R)	Bosch B5512 Control Panel (SKU: B5512)	Pelco IMP1110-1ES IP Camera (SKU: IMP1110-1ES)	GE Interlogix 60-807-95R Glassbreak Detector (SKU: 60-807-95R)	Bosch F220-B6 Detector Base (SKU: F220-B6)
2	3	Bosch F220-B6 Detector Base (SKU: F220-B6)	GE Interlogix 60-652-95R Carbon Monoxide Detector (SKU: 60-652-95R)	DSC WS4933 Carbon Monoxide Detector (SKU: WS4933)	NaN	NaN
3	4	GE Interlogix 60-652-95R Carbon Monoxide Detector (SKU: 60-652-95R)	Dahua 8-Channel NVR (SKU: NVR4208-8P-4KS2)	Bosch NDN-50022-A3 IP Camera (SKU: NDN-50022-A3)	Dahua DH-IPC-HDBW4431R-ZS IP Camera (SKU: DH-IPC-HDBW4431R-ZS)	Bosch B5512 Control Panel (SKU: B5512)
4	5	Hanwha QNV-6010R Network Camera (SKU: QNV-6010R)	Pelco DSSRV2-040-US NVR (SKU: DSSRV2-040-US)	DSC WS4916 Smoke Detector (SKU: WS4916)	Axis P3367-VE Network Camera (SKU: 0407-001)	Dahua DH-IPC-HDBW4431R-ZS IP Camera (SKU: DH-IPC-HDBW4431R-ZS)

```
/usr/local/lib/python3.12/dist-packages/jupyter_client/session.py:203: DeprecationWarning: datetime.datetime.utcnow() is deprecated and scheduled for removal in a future version. Use timezone-aware objects
  return datetime.utcnow().replace(tzinfo=utc)
/usr/local/lib/python3.12/dist-packages/jupyter_client/session.py:203: DeprecationWarning: datetime.datetime.utcnow() is deprecated and scheduled for removal in a future version. Use timezone-aware objects
```

🔄 Convert **wide → tall** (melt)

- Uses `transaction_id` as the transaction key
- Rows are melted from `item_1..item_5`
- Removes trailing `(SKU: …)` to reduce accidental uniqueness

[5]
✓ 0s

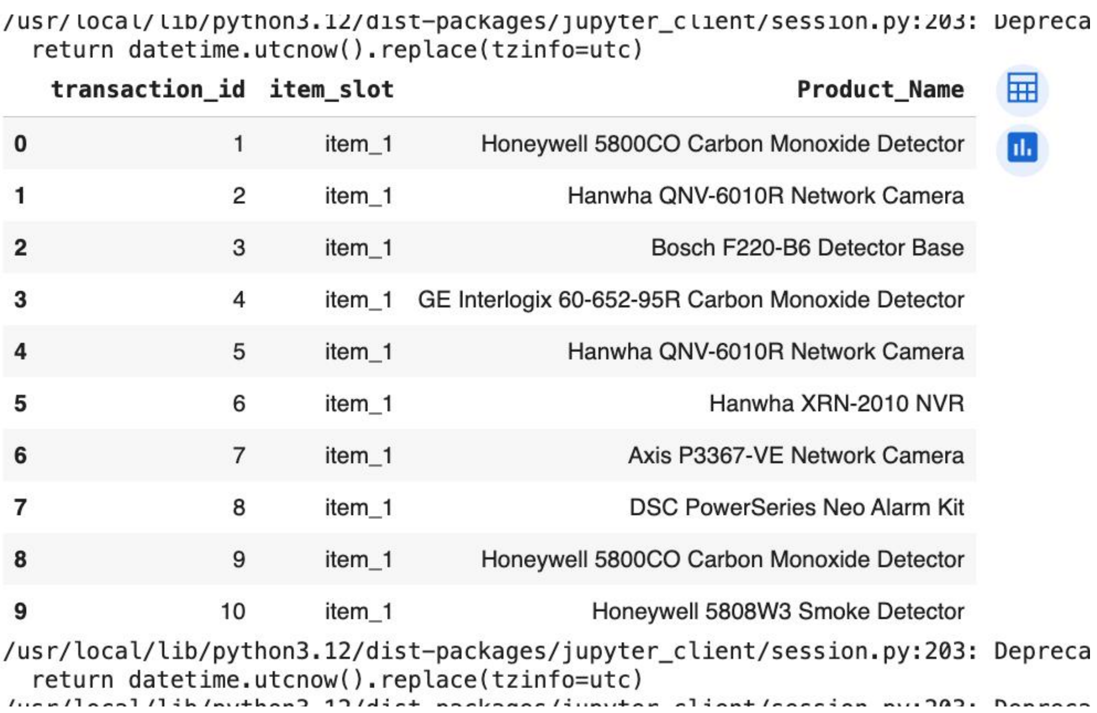
```python
# Validate required columns
required = ["transaction_id"] + ITEM_COLS
missing = [c for c in required if c not in df_raw.columns]
if missing:
    raise ValueError(f"Missing required columns for wide schema: {missing}\n"
                     f"Found: {list(df_raw.columns)}")

def normalize_item(x):
    x = str(x).strip()
    if not x or x.lower() == "nan":
        return ""
    x = re.sub(r"\s*\(SKU:\s*[^)]+\)\s*$", "", x)  # strip ' (SKU: ...)'
    x = re.sub(r"\s+", " ", x)
    return x.strip()

df_tall = df_raw.melt(
    id_vars=['transaction_id'],
    value_vars=ITEM_COLS,
    var_name='item_slot',
    value_name='Product_Name'
)
df_tall['Product_Name'] = df_tall['Product_Name'].astype(str).map(normalize_item)
df_tall = df_tall[df_tall['Product_Name'] != ""]

print("Tall shape:", df_tall.shape)
display(df_tall.head(10))
```

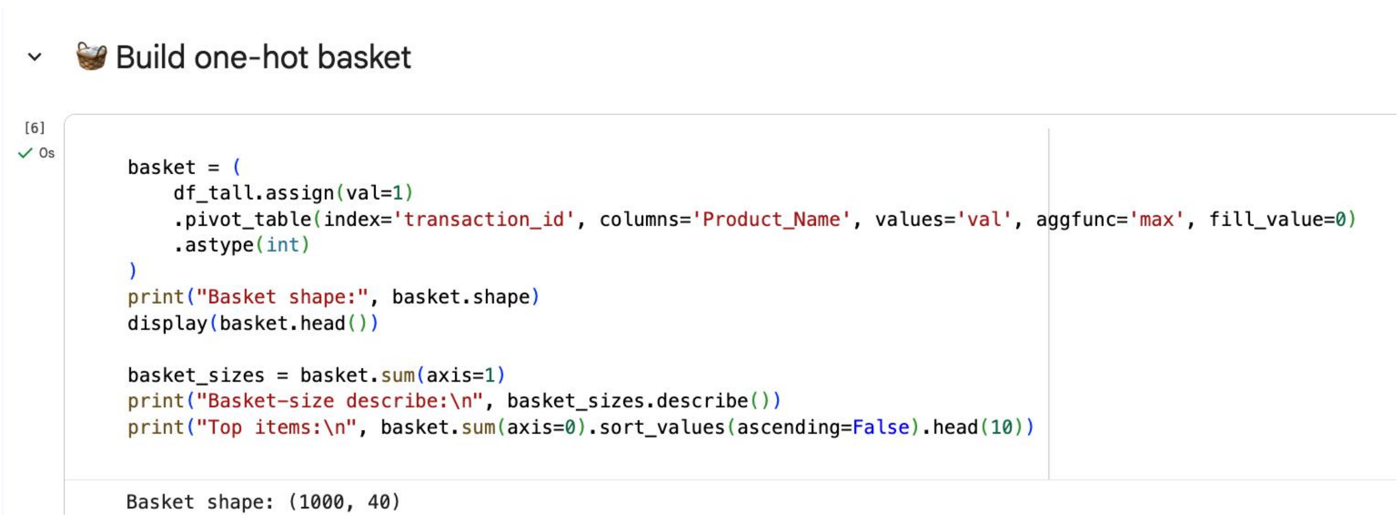
```
Tall shape: (3575, 3)
/usr/local/lib/python3.12/dist-packages/jupyter_client/session.py:203: DeprecationWarning:
  return datetime.utcnow().replace(tzinfo=utc)
```



```
/usr/local/lib/python3.12/dist-packages/jupyter_client/session.py:203: Depreca
  return datetime.utcnow().replace(tzinfo=utc)
```

	transaction_id	item_slot	Product_Name
0	1	item_1	Honeywell 5800CO Carbon Monoxide Detector
1	2	item_1	Hanwha QNV-6010R Network Camera
2	3	item_1	Bosch F220-B6 Detector Base
3	4	item_1	GE Interlogix 60-652-95R Carbon Monoxide Detector
4	5	item_1	Hanwha QNV-6010R Network Camera
5	6	item_1	Hanwha XRN-2010 NVR
6	7	item_1	Axis P3367-VE Network Camera
7	8	item_1	DSC PowerSeries Neo Alarm Kit
8	9	item_1	Honeywell 5800CO Carbon Monoxide Detector
9	10	item_1	Honeywell 5808W3 Smoke Detector

```
/usr/local/lib/python3.12/dist-packages/jupyter_client/session.py:203: Depreca
  return datetime.utcnow().replace(tzinfo=utc)
```



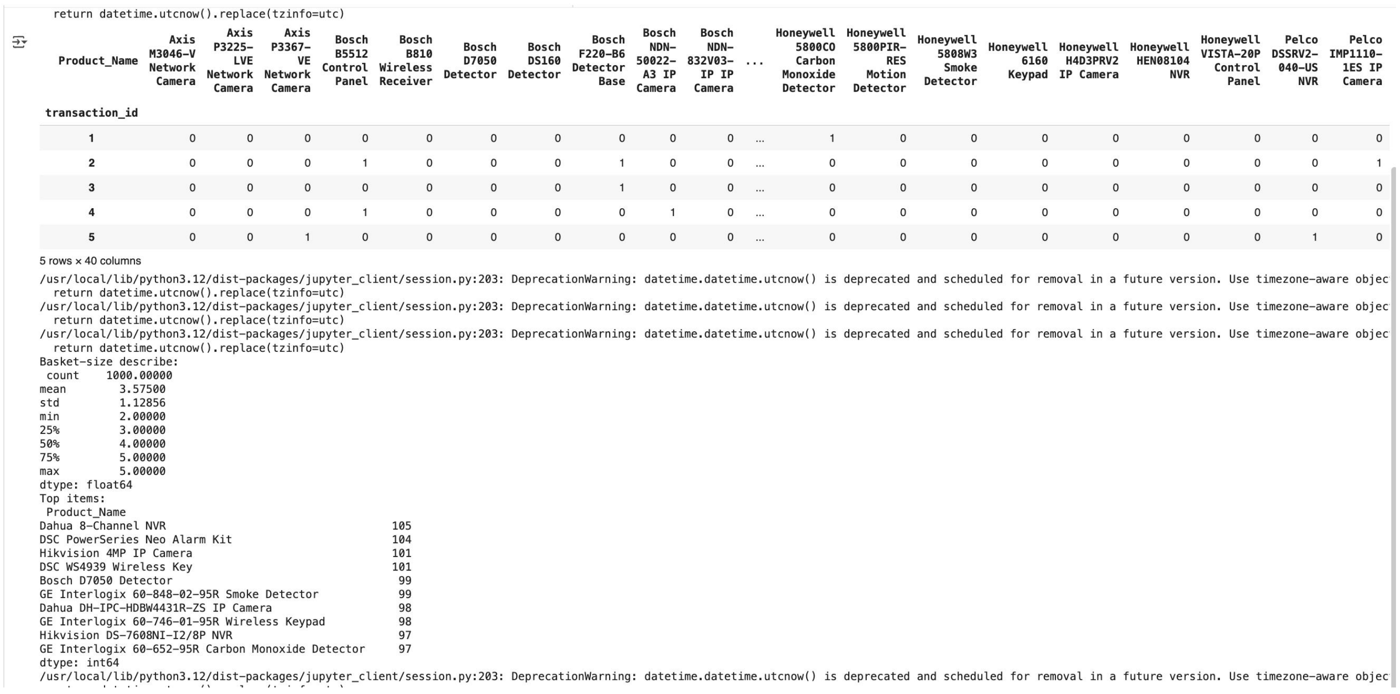
🧺 Build one-hot basket

[6] ✓ 0s

```python
basket = (
    df_tall.assign(val=1)
    .pivot_table(index='transaction_id', columns='Product_Name', values='val', aggfunc='max', fill_value=0)
    .astype(int)
)
print("Basket shape:", basket.shape)
display(basket.head())

basket_sizes = basket.sum(axis=1)
print("Basket-size describe:\n", basket_sizes.describe())
print("Top items:\n", basket.sum(axis=0).sort_values(ascending=False).head(10))
```

```
Basket shape: (1000, 40)
```



```
  return datetime.utcnow().replace(tzinfo=utc)
```

Product_Name	Axis M3046-V Network Camera	Axis P3225-LVE Network Camera	Axis P3367-VE Network Camera	Bosch B5512 Control Panel	Bosch B810 Wireless Receiver	Bosch D7050 Detector	Bosch DS160 Detector	Bosch F220-B6 Detector Base	Bosch NDN-50022-A3 IP Camera	Bosch NDN-832V03-IP IP Camera	...	Honeywell 5800CO Carbon Monoxide Detector	Honeywell 5800PIR-RES Motion Detector	Honeywell 5808W3 Smoke Detector	Honeywell 6160 Keypad	Honeywell H4D3PRV2 IP Camera	Honeywell HEN08104 NVR	Honeywell VISTA-20P Control Panel	Pelco DSSRV2-040-US NVR	Pelco IMP1110-1ES IP Camera
transaction_id																				
1	0	0	0	0	0	0	0	0	0	0	...	1	0	0	0	0	0	0	0	0
2	0	0	0	1	0	0	0	1	0	0	...	0	0	0	0	0	0	0	0	1
3	0	0	0	0	0	0	0	1	0	0	...	0	0	0	0	0	0	0	0	0
4	0	0	0	1	0	0	0	0	1	0	...	0	0	0	0	0	0	0	0	0
5	0	0	1	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	1	0

5 rows × 40 columns

```
/usr/local/lib/python3.12/dist-packages/jupyter_client/session.py:203: DeprecationWarning: datetime.datetime.utcnow() is deprecated and scheduled for removal in a future version. Use timezone-aware objec
  return datetime.utcnow().replace(tzinfo=utc)
/usr/local/lib/python3.12/dist-packages/jupyter_client/session.py:203: DeprecationWarning: datetime.datetime.utcnow() is deprecated and scheduled for removal in a future version. Use timezone-aware objec
  return datetime.utcnow().replace(tzinfo=utc)
/usr/local/lib/python3.12/dist-packages/jupyter_client/session.py:203: DeprecationWarning: datetime.datetime.utcnow() is deprecated and scheduled for removal in a future version. Use timezone-aware objec
  return datetime.utcnow().replace(tzinfo=utc)
Basket-size describe:
 count    1000.00000
mean        3.57500
std         1.12856
min         2.00000
25%         3.00000
50%         4.00000
75%         5.00000
max         5.00000
dtype: float64
Top items:
 Product_Name
Dahua 8-Channel NVR                               105
DSC PowerSeries Neo Alarm Kit                     104
Hikvision 4MP IP Camera                           101
DSC WS4939 Wireless Key                           101
Bosch D7050 Detector                               99
GE Interlogix 60-848-02-95R Smoke Detector         99
Dahua DH-IPC-HDBW4431R-ZS IP Camera                98
GE Interlogix 60-746-01-95R Wireless Keypad        98
Hikvision DS-7608NI-I2/8P NVR                      97
GE Interlogix 60-652-95R Carbon Monoxide Detector   97
dtype: int64
/usr/local/lib/python3.12/dist-packages/jupyter_client/session.py:203: DeprecationWarning: datetime.datetime.utcnow() is deprecated and scheduled for removal in a future version. Use timezone-aware objec
```

📈 Apriori rules (auto-tuned support)

Tries several supports until rules appear.

[7]
✓ 0s

```python
def build_rules_with_autotune(basket, min_lift=0.8):
    N = basket.shape[0]
    supports = [0.05, 0.02, 0.01, 0.005, 0.002, 0.001]
    supports += [max(1, int(0.005*N))/N, max(1, int(0.001*N))/N, 1/max(N,1)]
    tried = []
    for s in supports:
        s = float(s)
        if s <= 0 or s > 1:
            continue
        fi = apriori(basket, min_support=s, use_colnames=True)
        if fi.empty:
            tried.append((s, 0, 0))
            continue
        rules = association_rules(fi, metric="lift", min_threshold=0.0)
        if rules.empty:
            tried.append((s, len(fi), 0))
            continue
        rules = rules[rules["lift"] >= min_lift].copy()
        if rules.empty:
            tried.append((s, len(fi), 0))
            continue
        to_list = lambda x: list(x) if not isinstance(x, list) else x
        rules["antecedents"] = rules["antecedents"].apply(to_list)
        rules["consequents"]  = rules["consequents"].apply(to_list)
        print(f"✅ Rules found with min_support={s:.6f} (itemsets={len(fi)}, rules={len(rules)})")
        return rules.sort_values(["lift","confidence"], ascending=False).reset_index(drop=True), s
    print("⚠️ No strong rules found. Tried supports:", tried)
    return pd.DataFrame(), None

rules, used_support = build_rules_with_autotune(basket, min_lift=MIN_LIFT)
print("Rules:", len(rules))
display(rules.head(10))
```

```
  return datetime.utcnow().replace(tzinfo=utc)
✅ Rules found with min_support=0.010000 (itemsets=147, rules=214)
Rules: 214
/usr/local/lib/python3.12/dist-packages/jupyter_client/session.py:203: DeprecationWarning: datetime.datetime.utcnow() is deprecated and scheduled for removal in a future version. Use timezone-aware objec
  return datetime.utcnow().replace(tzinfo=utc)
/usr/local/lib/python3.12/dist-packages/jupyter_client/session.py:203: DeprecationWarning: datetime.datetime.utcnow() is deprecated and scheduled for removal in a future version. Use timezone-aware objec
  return datetime.utcnow().replace(tzinfo=utc)
/usr/local/lib/python3.12/dist-packages/jupyter_client/session.py:203: DeprecationWarning: datetime.datetime.utcnow() is deprecated and scheduled for removal in a future version. Use timezone-aware objec
  return datetime.utcnow().replace(tzinfo=utc)
/usr/local/lib/python3.12/dist-packages/jupyter_client/session.py:203: DeprecationWarning: datetime.datetime.utcnow() is deprecated and scheduled for removal in a future version. Use timezone-aware objec
  return datetime.utcnow().replace(tzinfo=utc)
/usr/local/lib/python3.12/dist-packages/jupyter_client/session.py:203: DeprecationWarning: datetime.datetime.utcnow() is deprecated and scheduled for removal in a future version. Use timezone-aware objec
  return datetime.utcnow().replace(tzinfo=utc)
```

	antecedents	consequents	antecedent support	consequent support	support	confidence	lift	representativity	leverage	conviction	zhangs_metric	jaccard	certainty	kulczynski
0	[Honeywell 5800CO Carbon Monoxide Detector]	[DSC WS4916 Smoke Detector]	0.075	0.089	0.015	0.200000	2.247191	1.0	0.008325	1.138750	0.600000	0.100671	0.121844	0.184270
1	[DSC WS4916 Smoke Detector]	[Honeywell 5800CO Carbon Monoxide Detector]	0.089	0.075	0.015	0.168539	2.247191	1.0	0.008325	1.112500	0.609221	0.100671	0.101124	0.184270
2	[Bosch NDN-832V03-IP IP Camera]	[Bosch NVR-5500-16A00 NVR]	0.081	0.090	0.014	0.172840	1.920439	1.0	0.006710	1.100149	0.521530	0.089172	0.091032	0.164198
3	[Bosch NVR-5500-16A00 NVR]	[Bosch NDN-832V03-IP IP Camera]	0.090	0.081	0.014	0.155556	1.920439	1.0	0.006710	1.088289	0.526688	0.089172	0.081127	0.164198
4	[Pelco DSSRV2-040-US NVR]	[GE Interlogix 60-807-95R Glassbreak Detector]	0.080	0.082	0.012	0.150000	1.829268	1.0	0.005440	1.080000	0.492754	0.080000	0.074074	0.148171
5	[GE Interlogix 60-807-95R Glassbreak Detector]	[Pelco DSSRV2-040-US NVR]	0.082	0.080	0.012	0.146341	1.829268	1.0	0.005440	1.077714	0.493827	0.080000	0.072110	0.148171
6	[Axis M3046-V Network Camera]	[Pelco IMP1110-1ES IP Camera]	0.080	0.084	0.012	0.150000	1.785714	1.0	0.005280	1.077647	0.478261	0.078947	0.072052	0.146429
7	[Pelco IMP1110-1ES IP Camera]	[Axis M3046-V Network Camera]	0.084	0.080	0.012	0.142857	1.785714	1.0	0.005280	1.073333	0.480349	0.078947	0.068323	0.146429
8	[Dahua DH-IPC-HFW4431R-Z IP Camera]	[Axis P3367-VE Network Camera]	0.085	0.090	0.013	0.152941	1.699346	1.0	0.005350	1.074306	0.449769	0.080247	0.069166	0.148693
9	[Axis P3367-VE Network Camera]	[Dahua DH-IPC-HFW4431R-Z IP Camera]	0.090	0.085	0.013	0.144444	1.699346	1.0	0.005350	1.069481	0.452240	0.080247	0.064967	0.148693

🔁 Co-occurrence fallback

Item–item counts ensure the UI still gives suggestions when rules are sparse.

[8]
✓ 0s

```python
co = basket.T.dot(basket)
for i in range(len(co)):
    co.iat[i, i] = 0
pop = basket.sum(axis=0)

def recommend_apriori(cart_items, top_k=TOP_K, rules_df=rules):
    if rules_df is None or rules_df.empty or not cart_items:
        return []
    cart_set = set(cart_items)
    cands = []
    for _, r in rules_df.iterrows():
        ant = set(r["antecedents"])
        if ant and ant.issubset(cart_set):
            for c in r["consequents"]:
                if c not in cart_set:
                    cands.append((c, float(r.get("confidence",0)), float(r.get("lift",0))))
    if not cands:
        return []
    ranked = sorted(cands, key=lambda x: (-x[2], -x[1], x[0]))
    out, seen = [], set()
    for item, conf, lift in ranked:
        if item not in seen:
            out.append(f"{item}  (lift={lift:.2f}, conf={conf:.2f})")
            seen.add(item)
        if len(out) >= top_k:
            break
    return out
```

```python
def recommend_cooccurrence(cart_items, top_k=TOP_K):
    if not cart_items:
        return ["(Select items first)"]
    valid = [i for i in cart_items if i in co.index]
    if not valid:
        return ["(Selected items not in matrix — check data)"]
    scores = (co[valid].sum(axis=1)).sort_values(ascending=False)
    scores = scores[~scores.index.isin(valid)]
    if len(scores)==0 or scores.max()==0:
        return ["(no strong co-occurrence found — please inspect data)"]
    s = pd.concat([scores.rename("score"), pop.rename("pop")], axis=1).fillna(0)
    s = s.sort_values(by=["score","pop"], ascending=False).head(top_k)
    return [f"{idx}  (co_count={int(row['score'])}, pop={int(row['pop'])})" for idx,row in s.iterrows()]

def recommend_with_fallback(cart_items, top_k=TOP_K):
    via_rules = recommend_apriori(cart_items, top_k=top_k, rules_df=rules)
    if via_rules:
        return via_rules
    return recommend_cooccurrence(cart_items, top_k=top_k)
```

🖥️ Gradio UI

Select items (multi-select) → **Get Recommendations**.

[9]
✓ 2s

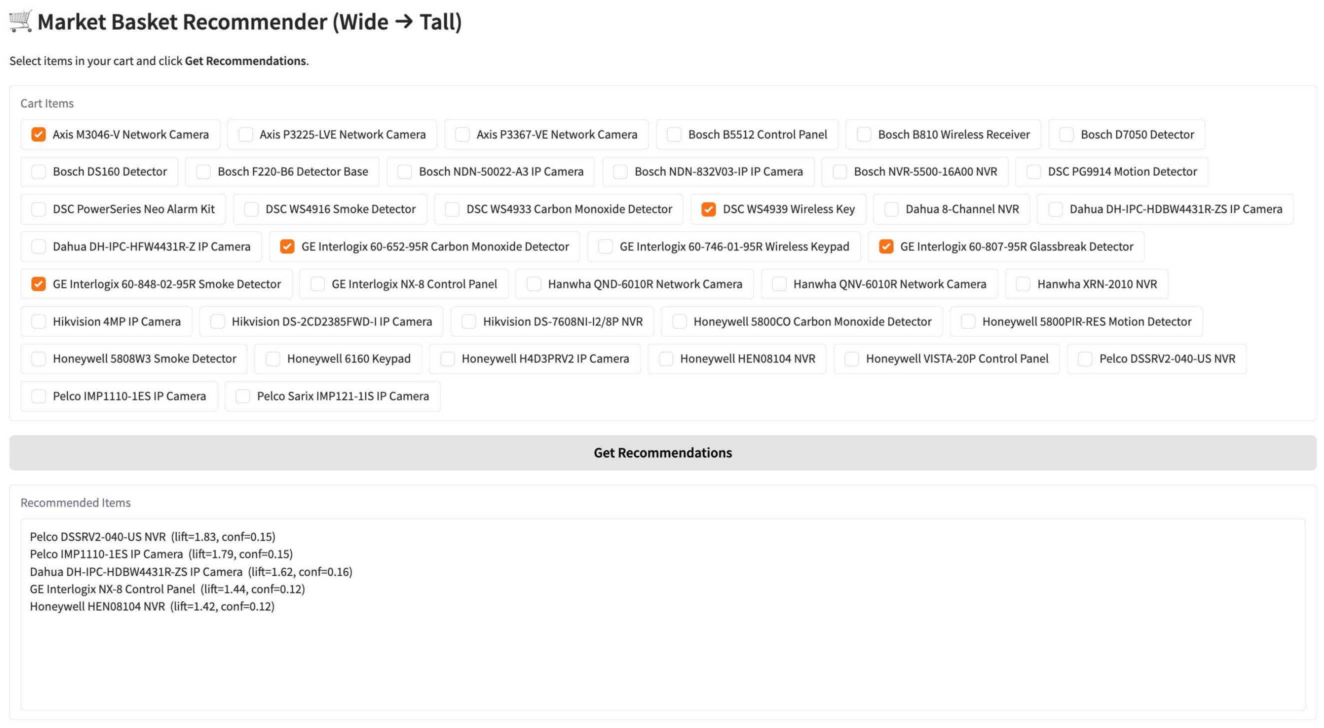
```python
items = sorted(list(basket.columns))

with gr.Blocks(title="Market Basket Recommender") as demo:
    gr.Markdown("# 🛒 Market Basket Recommender (Wide → Tall)")
    gr.Markdown("Select items in your cart and click **Get Recommendations**.")
    cart = gr.CheckboxGroup(choices=items, label="Cart Items")
    btn = gr.Button("Get Recommendations")
    out = gr.Textbox(label="Recommended Items", lines=10)

    def _go(selected):
        return "\n".join(recommend_with_fallback(selected or []))

    btn.click(_go, inputs=cart, outputs=out)

demo.launch(share=True)
```



🛒 Market Basket Recommender (Wide → Tall)

Select items in your cart and click **Get Recommendations**.

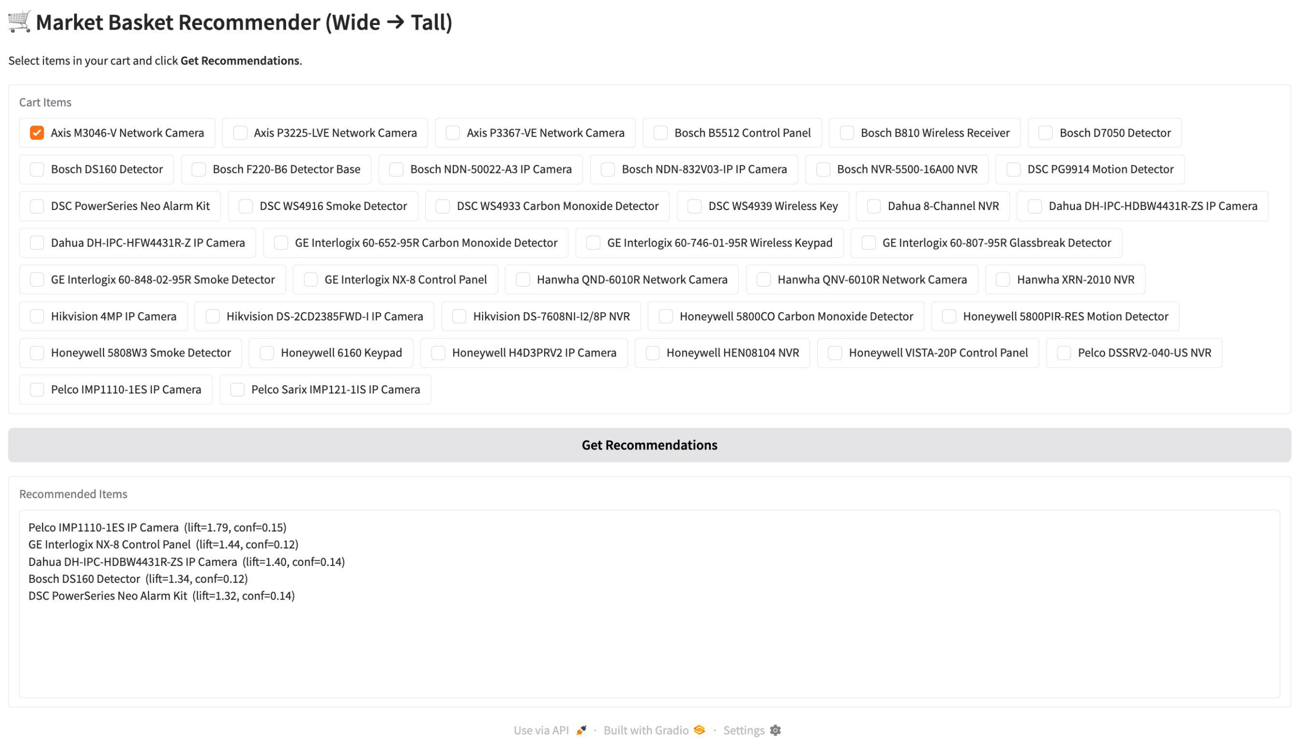
Cart Items

[✓ Axis M3046-V Network Camera] Axis P3225-LVE Network Camera | Axis P3367-VE Network Camera | Bosch B5512 Control Panel | Bosch B810 Wireless Receiver | Bosch D7050 Detector
Bosch DS160 Detector | Bosch F220-B6 Detector Base | Bosch NDN-50022-A3 IP Camera | Bosch NDN-832V03-IP IP Camera | Bosch NVR-5500-16A00 NVR | DSC PG9914 Motion Detector
DSC PowerSeries Neo Alarm Kit | DSC WS4916 Smoke Detector | DSC WS4933 Carbon Monoxide Detector | [✓ DSC WS4939 Wireless Key] Dahua 8-Channel NVR | Dahua DH-IPC-HDBW4431R-ZS IP Camera
Dahua DH-IPC-HFW4431R-Z IP Camera | [✓ GE Interlogix 60-652-95R Carbon Monoxide Detector] GE Interlogix 60-746-01-95R Wireless Keypad | [✓ GE Interlogix 60-807-95R Glassbreak Detector]
[✓ GE Interlogix 60-848-02-95R Smoke Detector] GE Interlogix NX-8 Control Panel | Hanwha QND-6010R Network Camera | Hanwha QNV-6010R Network Camera | Hanwha XRN-2010 NVR
Hikvision 4MP IP Camera | Hikvision DS-2CD2385FWD-I IP Camera | Hikvision DS-7608NI-I2/8P NVR | Honeywell 5800CO Carbon Monoxide Detector | Honeywell 5800PIR-RES Motion Detector
Honeywell 5808W3 Smoke Detector | Honeywell 6160 Keypad | Honeywell H4D3PRV2 IP Camera | Honeywell HEN08104 NVR | Honeywell VISTA-20P Control Panel | Pelco DSSRV2-040-US NVR
Pelco IMP1110-1ES IP Camera | Pelco Sarix IMP121-1IS IP Camera

Get Recommendations

Recommended Items

```
Pelco DSSRV2-040-US NVR  (lift=1.83, conf=0.15)
Pelco IMP1110-1ES IP Camera  (lift=1.79, conf=0.15)
Dahua DH-IPC-HDBW4431R-ZS IP Camera  (lift=1.62, conf=0.16)
GE Interlogix NX-8 Control Panel  (lift=1.44, conf=0.12)
Honeywell HEN08104 NVR  (lift=1.42, conf=0.12)
```


🛒 Market Basket Recommender (Wide → Tall)

Select items in your cart and click **Get Recommendations**.

Cart Items

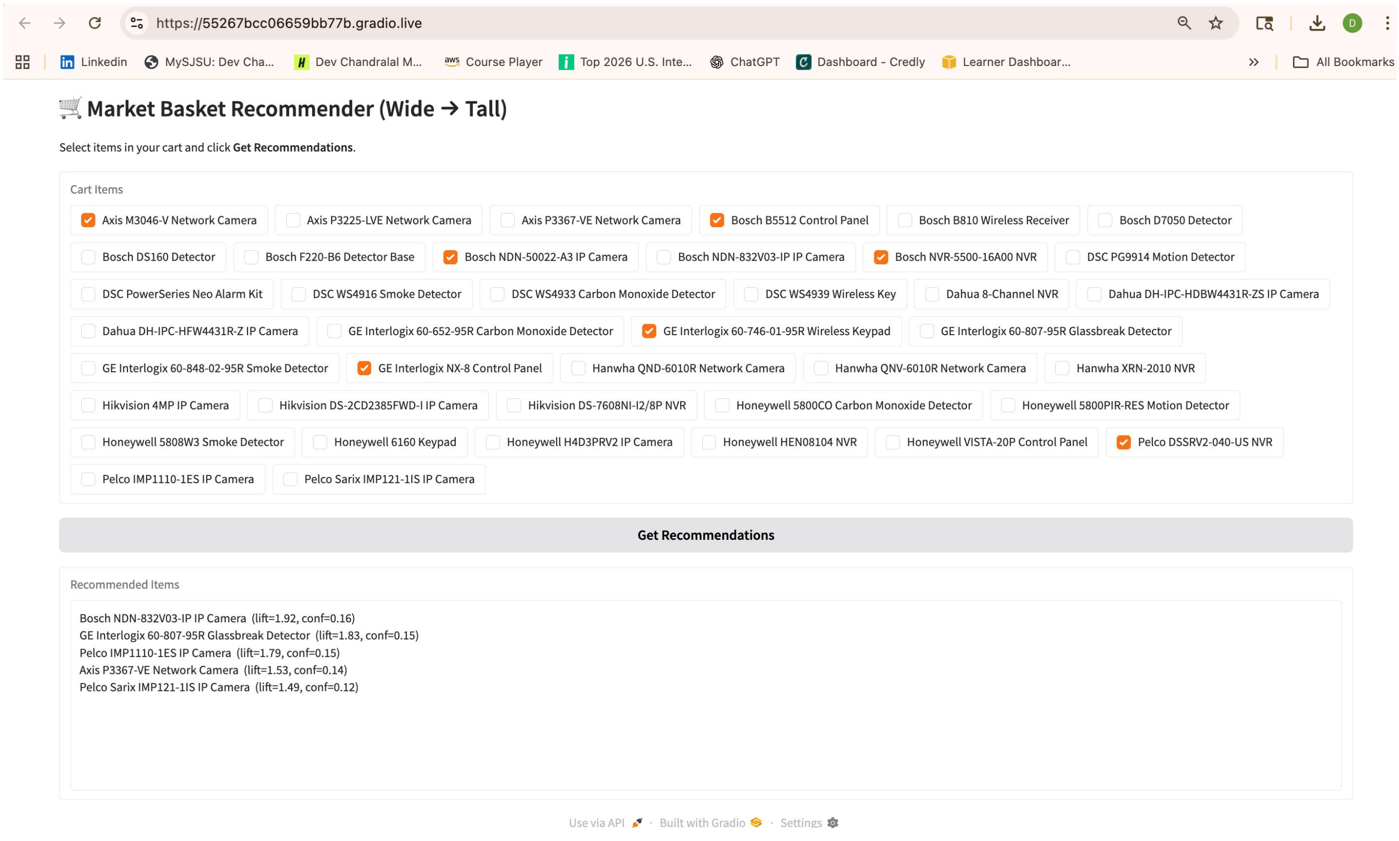
[✓ Axis M3046-V Network Camera] Axis P3225-LVE Network Camera · Axis P3367-VE Network Camera · Bosch B5512 Control Panel · Bosch B810 Wireless Receiver · Bosch D7050 Detector · Bosch DS160 Detector · Bosch F220-B6 Detector Base · Bosch NDN-50022-A3 IP Camera · Bosch NDN-832V03-IP IP Camera · Bosch NVR-5500-16A00 NVR · DSC PG9914 Motion Detector · DSC PowerSeries Neo Alarm Kit · DSC WS4916 Smoke Detector · DSC WS4933 Carbon Monoxide Detector · DSC WS4939 Wireless Key · Dahua 8-Channel NVR · Dahua DH-IPC-HDBW4431R-ZS IP Camera · Dahua DH-IPC-HFW4431R-Z IP Camera · GE Interlogix 60-652-95R Carbon Monoxide Detector · GE Interlogix 60-746-01-95R Wireless Keypad · GE Interlogix 60-807-95R Glassbreak Detector · GE Interlogix 60-848-02-95R Smoke Detector · GE Interlogix NX-8 Control Panel · Hanwha QND-6010R Network Camera · Hanwha QNV-6010R Network Camera · Hanwha XRN-2010 NVR · Hikvision 4MP IP Camera · Hikvision DS-2CD2385FWD-I IP Camera · Hikvision DS-7608NI-I2/8P NVR · Honeywell 5800CO Carbon Monoxide Detector · Honeywell 5800PIR-RES Motion Detector · Honeywell 5808W3 Smoke Detector · Honeywell 6160 Keypad · Honeywell H4D3PRV2 IP Camera · Honeywell HEN08104 NVR · Honeywell VISTA-20P Control Panel · Pelco DSSRV2-040-US NVR · Pelco IMP1110-1ES IP Camera · Pelco Sarix IMP121-1IS IP Camera

Get Recommendations

Recommended Items

```
Pelco IMP1110-1ES IP Camera  (lift=1.79, conf=0.15)
GE Interlogix NX-8 Control Panel  (lift=1.44, conf=0.12)
Dahua DH-IPC-HDBW4431R-ZS IP Camera  (lift=1.40, conf=0.14)
Bosch DS160 Detector  (lift=1.34, conf=0.12)
DSC PowerSeries Neo Alarm Kit  (lift=1.32, conf=0.14)
```

Use via API · Built with Gradio · Settings


https://55267bcc06659bb77b.gradio.live

Linkedin · MySJSU: Dev Cha... · Dev Chandralal M... · Course Player · Top 2026 U.S. Inte... · ChatGPT · Dashboard - Credly · Learner Dashboar... · All Bookmarks

🛒 Market Basket Recommender (Wide → Tall)

Select items in your cart and click **Get Recommendations**.

Cart Items

[✓ Axis M3046-V Network Camera] Axis P3225-LVE Network Camera · Axis P3367-VE Network Camera · [✓ Bosch B5512 Control Panel] Bosch B810 Wireless Receiver · Bosch D7050 Detector · Bosch DS160 Detector · Bosch F220-B6 Detector Base · [✓ Bosch NDN-50022-A3 IP Camera] Bosch NDN-832V03-IP IP Camera · [✓ Bosch NVR-5500-16A00 NVR] DSC PG9914 Motion Detector · DSC PowerSeries Neo Alarm Kit · DSC WS4916 Smoke Detector · DSC WS4933 Carbon Monoxide Detector · DSC WS4939 Wireless Key · Dahua 8-Channel NVR · Dahua DH-IPC-HDBW4431R-ZS IP Camera · Dahua DH-IPC-HFW4431R-Z IP Camera · GE Interlogix 60-652-95R Carbon Monoxide Detector · [✓ GE Interlogix 60-746-01-95R Wireless Keypad] GE Interlogix 60-807-95R Glassbreak Detector · GE Interlogix 60-848-02-95R Smoke Detector · [✓ GE Interlogix NX-8 Control Panel] Hanwha QND-6010R Network Camera · Hanwha QNV-6010R Network Camera · Hanwha XRN-2010 NVR · Hikvision 4MP IP Camera · Hikvision DS-2CD2385FWD-I IP Camera · Hikvision DS-7608NI-I2/8P NVR · Honeywell 5800CO Carbon Monoxide Detector · Honeywell 5800PIR-RES Motion Detector · Honeywell 5808W3 Smoke Detector · Honeywell 6160 Keypad · Honeywell H4D3PRV2 IP Camera · Honeywell HEN08104 NVR · Honeywell VISTA-20P Control Panel · [✓ Pelco DSSRV2-040-US NVR] Pelco IMP1110-1ES IP Camera · Pelco Sarix IMP121-1IS IP Camera

Get Recommendations

Recommended Items

```
Bosch NDN-832V03-IP IP Camera  (lift=1.92, conf=0.16)
GE Interlogix 60-807-95R Glassbreak Detector  (lift=1.83, conf=0.15)
Pelco IMP1110-1ES IP Camera  (lift=1.79, conf=0.15)
Axis P3367-VE Network Camera  (lift=1.53, conf=0.14)
Pelco Sarix IMP121-1IS IP Camera  (lift=1.49, conf=0.12)
```

Use via API · Built with Gradio · Settings