

## DSO 560 Group Project Part II

**Due 11:59pm PST, Monday May 11<sup>th</sup>**

All available data will be uploaded to the **threadtogether** PostgreSQL database, available at

Host: **threadtogether.ychennay.com**

User: **dso560\_student**

Password: **(in class)**

Port: **5432**

**If your group would like to use the database to create temporary tables, please message me and I will create a group-specific user/password for your team to log in as.**

### **Deliverables:**

- **A notebook** that allows a user to enter inputs (defined below) – either product IDs or product descriptions and details, and returns recommended outfits.
- **Submit this code to the same Github repository** (with instructor ychennay added as collaborator containing your group's product attribution tool/app code base) you uploaded for Part I.

### **Database Tables:**

- **outfit\_combinations\_view:** a view lists outfits curated by the subject matter expert (SME) stylist:
  - **outfit\_id:** identifies the outfit – a combination of product IDs
  - **product\_id:** the product ID of the piece of clothing
  - **outfit\_item\_type:** the part of the outfit the recommendation is for

The view is also available as a CSV file here: **[https://dso-560-nlp-text-analytics.s3.amazonaws.com/outfit\\_combinations.csv](https://dso-560-nlp-text-analytics.s3.amazonaws.com/outfit_combinations.csv)**

### **Objectives:**

**Outfit recommendations:** Based upon the **outfit\_combinations** view, build a model/algorithm that will accept 1 of the following inputs for outfit type:

- a shoe
- a bottom
- a top
- an accessory
- a one-piece

**And return a recommendation for an outfit for a “similar” aesthetic or fashion based upon the outfit collections generated by the subject matter expert (SME) stylist.**

The inputs can be either

A. **product IDs**

B. **Free form text: brand, brand category, details, and description** (like the input for Part I).

#### Example A

<b>INPUT (Product IDs – what I pass in to the model):</b> <ul style="list-style-type: none"><li>shoe: <b>Penelope Mid Cap Toe Pump</b> (01DMBRYVA2ZFDYRYY5TRQZJTBD)</li></ul>
<b>OUTPUT (Recommended Outfit Combination – what is returned from the model (highlighted yellow are its recommendations for the outfit):</b> <ul style="list-style-type: none"><li>shoe: <b>Penelope Mid Cap Toe Pump</b> (01DMBRYVA2ZFDYRYY5TRQZJTBD)</li><li>bottom: <b>Slim Knit Skirt</b> (01DMBRYVA2P5H24WK0HTK4R0A1)</li><li>accessory: <b>medium margaux leather satchel</b> (01DMBRYVA2S5T9W793F4CY41HE)</li><li>top: <b>Rib Mock Neck Tank</b> (01DMBRYVA2PEPWFTT7RMP5AA1T)</li></ul>

#### Example B

<b>INPUT (Product Descriptions):</b> <ul style="list-style-type: none"><li>bottom: <i>DESCRIPTION: slim fitting, straight leg pant with a center back zipper and slightly cropped leg – BRAND: Reformation</i></li></ul>
<b>OUTPUT (Recommended Outfit Combination):</b> <ul style="list-style-type: none"><li>bottom: <b>Marlon Pant</b> (01DPKMH0D252JKMAA27MFCT5GM)</li><li>shoe: <b>Giulia Satin Heel</b> (01DPKNHQDG6GPTKV97CFQRJDHE)</li><li>accessory: <b>Cassi Belt Bag</b> (01DPEHS0XH9PDD1GH5ZE4P43A2)</li></ul>

The model should give reasonable recommendations for similar products. For instance, if I provide in as input the following product description:

*Sexy silky, a-line mini skirt zipper Benson skirt*

The model should be able to identify this as a close match against **Product ID 01DPKMGJ33SDFXM7XHGPQJWQ12**, the **Benson skirt**, which has an actual product description of

*Sexy silky. This is an a-line mini skirt with a center back zipper. The Benson pairs well with the Hailee Top.*

and recommend this product ID as the outfit's **bottom**, and from there, return recommendations for a **top**, or an **accessory**, or **shoe** (at least two recommendations). ***Note: If you match a product that already exists in the outfit\_combinations, you can simply return the rest of its outfit recommendations from that same table.***

My test queries will be similar to these – I will pick a few products, change up their wording a bit, use some synonyms, use similar words, but expect that your model should be able to find a similar product.

Scoring Rubric	Points Available			
	1pt	2pts	3pts	4pts
<b>The model will be tested by running 4 test queries, and returns expected matches (like the example above).</b>	No results are returned	Basic results are returned but they do actually make sense and are not similar in any way to the product query.	3/4 test queries submitted with a free-form product description are returned with products that are actually “similar”.	4/4 test queries submitted with a free-form product description are returned with products that are actually “similar”.
<b>The model handles both product IDs and also production descriptions + brand names (free-form text) as inputs</b>	No (0 pts)		Yes (2 pts)	
<b>Code is documented with thought process easily visible to instructor/client to review</b>	No (0 pts)		Yes (2 pts)	
<b>Code is published to a Github repository with team members added as collaborators</b>	No (0 pt)		Yes (1 pt)	
<b>All group members have submitted 360 feedback reviews by deadline</b>	No (0 pt)		Yes (1 pt)	

