

Assessment Submission Cover Sheet

This Assessment Cover Sheet **must** be included on all Assessment submissions.

Assignment Title	Machine Learning, Assignment 2
Module	SPEC9270 2021-22
Student Name	Robert O’Sullivan
Student Number	C08345457
Programme	TU060/DS
Part-Time/Full-Time	Part Time
Year of Study (First Year, Second Year, etc)	Second Semester

Late Submissions: Assessment submitted after the deadline will have a late penalty applied.

Academic Integrity for assessment in TU Dublin Programmes

Each student is responsible for knowing and abiding by TU Dublin Academic Regulations and Policies. Any student in breach of these regulation/policies will be subject to action in accordance with the University’s procedures for breaches of assessment regulations. Please refer to the General Assessment Regulations at

- <https://tudublin.libguides.com/c.php?g=674049&p=4794713>
- <https://www.tudublinsu.ie/advice/exams/breachesofregulations/>

All students are expected to complete their courses/programmes in compliance with university regulations. No student shall engage in any activity that involves attempting to receive a grade by means other than honest effort, for example:

1. No student shall complete, in part or in total, any examination or assessment for another person.
2. No student shall knowingly allow any examination or assessment to be completed, in part or in total, for themselves by another person.
3. No student shall plagiarise or copy the work of another and submit it as their own work.
4. No student shall falsify any data. Falsification is the invention of data, its alteration, its copying from any other source, or otherwise obtaining it by unfair means, or inventing quotations and/or references.
5. No student shall use aids or devices excluded by the lecturer in undertaking course work or assessments/ examinations.
6. No student shall knowingly procure, provide, or accept any materials that contain questions or answers to any examination or assessment to be given at a subsequent time.
7. No student shall provide their assignments, in part or in total, to any other student in current or future classes of this module/ programme unless authorised to do so by the lecturer.
8. No student shall submit substantially the same material in more than one module/programme without prior authorization.
9. No student shall alter graded assignments or examinations and then resubmit them for regrading, unless specifically authorised to do so by the lecturer.
10. All programming code and documentation, unless correctly referenced, submitted for assessment or existing in the student’s computer accounts must be the students’ original work or material specifically authorized by the lecturer.
11. Collaborating with other students to develop, complete or correct course work is limited to activities explicitly authorized by the lecturer.
12. For all group assignments, each member of the group is responsible for the academic integrity of the entire submission. Consequently, all group members must satisfy themselves that all elements of their submission adhere to the academic integrity statement points above.

By submitting coursework, either physically or electronically, you are confirming that it is your own work (or, in the case of a group submission, that it is the result of joint work undertaken by members of the group that you represent) and that you have read and understand the University’s Regulations and Policies covering Academic Integrity (see General Assessment Regulations).

Coursework may be submitted to an electronic detection system in order to help ascertain if any plagiarised material is present. If you have queries about what constitutes plagiarism, please speak to your lecturer.

Student Signature	Robert O’Sullivan.
Date	06/05/2022

Table of Contents

Introduction	3
Related work	3
Summary & Identified Important Features.....	4
Exploratory Data Analysis (EDA)	4
Transaction Dataset	5
Customers Dataset.....	5
Articles Dataset	5
Summary of EDA and Model Chosen	6
Data Preparation.....	7
Preparing the Transaction Dataset	7
Creating Training and Testing Dataset.....	7
Training the Model.....	8
Evaluating the Model	8
Generating Kaggle Submission File	9
Conclusion.....	10
Feature Selection Opportunities.....	10
Dimension Reduction Opportunities	10
Memory Management Opportunities	10
Model Opportunities	10
References	11
Appendix	12
Equations	12
Figures.....	13

Introduction

H&M Group is a clothing company with 53 online markets and 4,850 stores. H&M customers may not find products that interests them resulting in no purchase. Customer returns also increase H&M's CO2 emissions and impact the environment. H&M want a product recommender based on data from previous transactions, as well as from customer and product meta data. This report summarises related research in the field of fashion recommendation, exploratory data analysis, data preparation, model training and testing of a K-Nearest Neighbour (KNN) machine learning model and how a file was generated and submitted to H&M via Kaggle for their evaluation.

H&M want a prediction file containing 12 products customers will buy in the next 7-day period regardless of whether they purchased from H&M in the past. This means any significant scoring and contribution will be impossible without sufficient memory and compute power. Other than an intel i5 CPU, 8gb Ram, 500 SSD windows laptop, no hardware was made available to complete this task. Free tier accounts from Google or Kaggle only provided a limited 12 hours of compute time. This will not be enough for model optimisation. A small sample of the 31.9 million transactions, 1.37 million customers and 105,542 products were made to preform local evaluation.

Related work

(Ko et al, 2022), states recommender systems are 1) Collaborative Filtering based. 2) Content based or 3) Hybrid based systems. Collaborative Filtering systems assume customers who liked similar items in the past will like similar items in the future. These systems suffer from the cold-start problem where not enough past information is available for each user on all products or for new users. Content-Based systems use data about users and products to make predictions, specifically, text mining, semantic analysis, Term Frequency Inverse Document Frequency (TF-IDF), Neural Network, Naive Bayes, and support vector machines (SVM). These systems suffer from sparsity, cold-start, and gray sheep problems. They also do not show customers new products. Popular CF model-based systems use clustering, SVD and PCA. Since 2012, research into Hybrid Recommendations have increased and decreased application of standalone systems. Precision, Recall, Accuracy, F-Measure, ROC Curve and AUC are common evaluation metrics for recommendation systems.

(Deldjoo et al, 2022) states fashion recommender systems attempt to recommend 1) items, 2) outfits and pairs, 3) clothing size recommendation or 4) provided improve explanations for recommendations. User-item preference scoring is done through Equation 1 – Utility Function. Equation 2 - Bayesian Personalised Ranking from implicit feedback (BPR-MF) is used, which is a simple pair-wise loss ranking model based off Equation 1, which has a Matrix Factorisation (MF) predictor and itself, forms the basis for many visual model-based recommendation systems. Equation 3 - Fashion Outfit Composition Score is used to create pairs and outfit recommendations. Variations exists to target customer demographic or model outfit sequences. Size is important if wrong customers will return the product. Better explanation could help convince users to purchase items faster. Explanation systems either describe internal model operations or create a post-hoc report. (Deldjoo et al, 2022) describe a system that exists, comprised of KMeans Clustering for colour and CNN for shapes to provide explanation to users. Popular Algorithms mostly consist of visually aware model-based CF systems. Equation 4 - Visual Bayesian Personalised Ranking (VBPR) is based on Equation 2 - Bayesian Personalised Ranking from implicit feedback (BPR-MF) but it combines convolutional neural networks to improve the BPR ranking. VBPR requires lots of image labelling but Equation 5 - DeepStyle formula focuses on learning clothing style from user-item matrix (style equal item minus category information). Image is an important feature for fashion recommendation systems, with a lot of research effort in computer vision. Progress towards Generative Fashion Recommendation Models

has been made to unearth hidden elements within user preferences. (Bellini et al, 2022) built a recommendation system for fashion retail shops based on a multi clustering approach of items and users' profiles in online and on physical stores increasing buyers' attention and purchase increase of 3.48%. (Ding et al, 2022) developed an attentional factor field interaction graph (AFFIG) to incorporate colour, style, brand, to predict implicit user item interaction. They used GPUs to train their NN models on. (Zhu & Van Roy, 2021) use reinforcement learning and simulations to build a recommendation system. (Mohammadi et al, 2021) used computer vision to develop a single product recommendation and developed a new scoring metric called objective-guided human score. In their thesis (Léa et al, 2018) concluded that social media influencers create a desire for products and faster purchase decision. This report explored how H&M customers were influenced to make purchasing decisions by searching YouTube and Instagram for H&M content. It found that content from these influences was promoted as a time series (spring fashion, fashion haul March 2022, etc.).

Summary & Identified Important Features

Most fashion recommendation formulas are based on user ratings and incorporate a visual dimension. Size is also important and if wrong customers will return the product. Fashion item representation, personalisation, sizing fit, interpretability, explanation and discovering trends remain major challenges facing fashion recommender systems with focus on computer vision research to solve these. We found that time of month was also important to influencers in creating a desire in customers to make purchasing decisions.

In this report we focused on model-based systems where we used transaction data to determine implicit interest to predict a recommendation. We were not given labelled data in the H&M dataset and image processing was computationally expensive given hardware available for processing. However, time of month, social membership, clothing size, colour, brand, visual features in images, texture style and textural features are considered important features for user and item representation. Since we do not actual have an explicit customer-item-ratings like a 1 to 5 rating per item, we will assume qty of purchase and level of price indicate implicit customer interest in products. Also, since the size of dataset given, we will try an unsupervised Nearest Neighbour approach to generate labels from the available data for classification. We will choose time and price as features to cluster for Nearest Neighbour to label to not suffer excess of features to process.

In the proceeding sections we will discuss how an unsupervised Nearest Neighbour model was chosen, built, and evaluated on the transaction dataset using day of the month and price level as features.

Exploratory Data Analysis (EDA)

An exploratory data analysis was conducted by various Kaggle contestants such as (Karpov, 2022) and (Licht Lab, 2022) to understand trends in the dataset. This report used EDA to determine the suitability of a machine learning model on data provided by H&M, help us understand what encoding requirements, understand dimensional reduction techniques available and evaluation criteria needed on this data.

Data available consisted of 1) image dataset - images of every product, 2) articles dataset - metadata of every product, 3) customer dataset - metadata of every customer and 4) transactions dataset - purchase details for customers who bought products. Although it was highlighted in Related work that images were an important part of how customers decide on the products they purchase, due to the data size and limited computer memory available, we did not use them, nor did we use any computer vision models.

Transaction, customer, and article datasets were all imported.

Transaction Dataset

There were over 31.9 million transactions available (3gb in size). With our limited space and processing power, this made working with the dataset slow and unwieldy. Instead, we only sampled the last two months of this dataset.

```
In [18]: transactions_train_df.head(3)
```

```
Out[18]:
```

	t_dat	customer_id	article_id	price	sales_channel_id
0	2018-09-20	000058a12d5b43e67d225668fa1f8d618c13dc232df0ca...	663713001	0.050831	2
1	2018-09-20	000058a12d5b43e67d225668fa1f8d618c13dc232df0ca...	541518023	0.030492	2
2	2018-09-20	00007d2de826758b65a93dd24ce629ed66842531df6699...	505221004	0.015237	2

```
In [13]: transactions_train_df.nunique()
```

```
Out[13]:
```

t_dat	734
customer_id	1362281
article_id	104547
price	9857
sales_channel_id	2
dtype:	int64

	year	month	day	price
count	3.178832e+07	3.178832e+07	3.178832e+07	3.178832e+07
mean	2.019207e+03	6.511067e+00	1.624134e+01	2.782927e-02
std	6.644412e-01	3.273328e+00	8.934254e+00	1.918113e-02
min	2.018000e+03	1.000000e+00	1.000000e+00	1.694915e-05
25%	2.019000e+03	4.000000e+00	8.000000e+00	1.581356e-02
50%	2.019000e+03	6.000000e+00	1.700000e+01	2.540678e-02
75%	2.020000e+03	9.000000e+00	2.400000e+01	3.388136e-02
max	2.020000e+03	1.200000e+01	3.100000e+01	5.915254e-01

Figure 1 distribution and unique values in transaction dataset

Customers Dataset

In the customer dataset we had 1.37 million customers from 352,899 locations, it was assumed that FN stood for whether the customers signed up for fashion news. NaNs will need to be converted into zero values for the FN and for customer member status, we will need to do the same for Active. Fashion news frequency will need original categories encoded.

```
In [14]: customers_df.head(3)
```

```
Out[14]:
```

	customer_id	FN	Active	club_member_status	fashion_news_frequency	age
0	00000dbacae5abe5e23885899a1fa44253a17956cd1c3...	NaN	NaN	ACTIVE	NONE	49.0
1	0000423b00ade91418cceaf3b26c6af3dd342b51fd051e...	NaN	NaN	ACTIVE	NONE	25.0
2	000058a12d5b43e67d225668fa1f8d618c13dc232df0ca...	NaN	NaN	ACTIVE	NONE	24.0

```
In [15]: customers_df.nunique()
```

```
Out[15]:
```

customer_id	1371980
FN	1
Active	1
club_member_status	3
fashion_news_frequency	4
age	84
postal_code	352899
dtype:	int64

Figure 2 unique values in customer dataset

Articles Dataset

105,542 products were in the articles dataset. Regarding columns in this dataset, every item had a unique identifier called the article_id.

The product name can be dropped as the product code and name seem to match. This is true for the product type name, colour group name and graphical appearance name. The 'product group name' would need to be kept and encoded as there was no corresponding number column associated with it.

article_id	product_code	prod_name	product_type_no	product_type_name	product_group_name	graphical_appearance_no	graphical_appearance_name	colour_group_code	colour_group_name	department_name	index_code	index_name	index_group_no	index_group_name	section_no	section_name	garment_group_no	garment_group_name	detail_desc
100027	952430001	SRK regular placement1	307	Socks	Socks & Tights	1210214	Plazacore print	5	Black	Socks 4in	P	Unicolor	3	Unicolor	26	Men Underwear	1001	Socks and Tights	Section in a 10-piece cotton blend with a white.
100038	952430001	SRK regular placement1	303	Vest top	Garment Upper body	1210216	Solid	5	Black	Jersey	A	1 adicolor	1	1 adicolor	2	H&M	1005	Jersey Knit	Loose fitting sports vest with a round fast.
100039	952430001	SRK regular placement1	303	Vest top	Garment Full body	1210216	Solid	5	Black	Jersey	A	1 adicolor	1	1 adicolor	18	Women's T-shirt	1005	Jersey Knit	Short & line dress in jersey with a round neck.
100040	952430001	CLARKS HAIR CLIP	2	Hair clip	Accessories	1210216	Solid	5	Black	Small Accessories	U	Unicolor	2	Unicolor	12	Divided Accessories	1009	Accessories	Linear elastic hair clip.
100041	952430001	SRK regular placement1	215	Dress	Garment Full body	1210216	Solid	11	Off White	Jersey	A	Ladieswear	1	Ladieswear	18	Women's T-shirt	1005	Jersey Knit	Full length dress in solid jersey with a round neck.

5 rows × 20 columns

5 rows x 25 columns

```
In [17]: articles_df.nunique()

Out[17]:
article_id          105542
product_code        47224
prod_name           45875
product_type_no      132
product_type_name    131
product_group_name    19
graphical_appearance_no  30
graphical_appearance_name  30
colour_group_code     50
colour_group_name     50
perceived_colour_value_id  8
perceived_colour_value_name  8
perceived_colour_master_id  20
perceived_colour_master_name  20
department_no        299
department_name       250
index_code           10
index_name            10
index_group_no        5
index_group_name       5
section_no           57
section_name          56
garment_group_no      21
garment_group_name     21
detail_desc          43404
dtype: int64
```

Figure 3 - unique values in articles dataset

Summary of EDA and Model Chosen

31.9 million transactions with 1,362,281 customers and 104,547 products highlights that some products and customers are missing. (Ko et al, 2022) and (Deldjoo et al, 2022) identify that K nearest neighbour is a starting point for recommendations, in this report an unsupervised KNN approach was chosen since we lack labelled data. Since KNN performs poorly when more features are added and from observations of influencers on social media based on (Léa et al, 2018) findings, days of the month and price will be used.

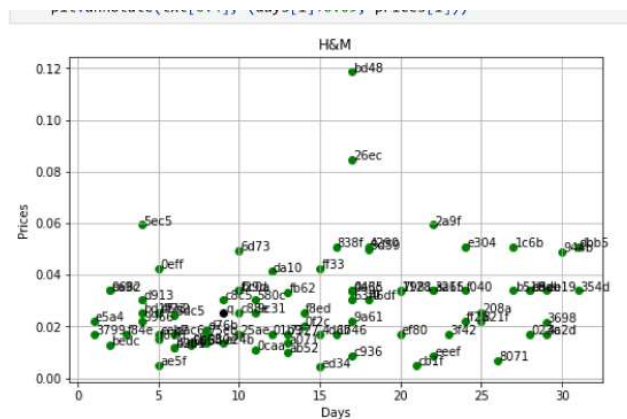


Figure 4 scatter plot of price and days of the month with a September (q) value placed with August values (X)

KNN will represent days and price as instance in a feature space then measure the similarity between instances. Distance metrics can be Euclidean, Manhattan, etc. This report used the default Euclidean distance, making the feature space a Euclidean coordinate space. Using Euclidean distance, according to (Kelleher et al, 2020), will give weight to one large difference in features rather than features with smaller differences. A sample of the large transaction database will be taken allowing us to use Euclidean. Otherwise, we would use Manhattan as it has a slight computational advantage but is influenced by small feature differences.

Care was when selecting a k value as KNN is sensitive to imbalanced data we started with 3 but increased it to 4 (i.e radius). During this data collection period some customers did not buy anything, and some products remained not selected. We do not have missing dates or prices. 68% of customers transactions lie within 1 standard deviation from the mean with 3 standard deviations being rare events (1%). The outliers are small and could possibly be related to customers buying for charities or clubs, or reselling on. Customer and articles dataset contain some missing information that will need to be corrected.

Data Preparation

Preparing the Transaction Dataset

Transaction dataset was imported then sales_channel_id dropped. t_dat was set to datetime. python zip and datetime functions were used to split the date into a year, month, and day column. Once finished with t_dat, it was dropped. Article_id was set to a string. The transaction dataset was then reorganised to have customer as the predictor and article_id set as index.

Creating Training and Testing Dataset

A sample of the transactions dataset was created that consisted of 50 samples from both august and September 2020 (the most recent transactions in the dataset).

```
aug_df = transactions_train_df.query('year == 2020 & month == 8').sample(n = 50)
sep_df = transactions_train_df.query('year == 2020 & month == 9').sample(n = 50)

sample_df = pd.concat([aug_df, sep_df])

#we set articles column to index
sample_df.set_index('article_id')
```

	year	month	day	price	customer_id
781833008	2020	8	26	0.006763	80715f6e81ebc23f27308e2ddde918aee046a6e6bce7dc...
706016003	2020	8	2	0.033881	ca9cdf6db1fed23dae0b8b4951d364b1ec8635b36dd901...
505882002	2020	8	14	0.025407	f8ed98a19fbd5ef24a09c221dc8b36023a287aa64beb85...
748355002	2020	8	24	0.022017	ff2ac6b20b6e6cc42dda998278e31c5bea53de2fb64389...
736870001	2020	8	20	0.016932	ef8073b39859baaf5fe3884cc3d8f516afb73f683f6b9...
...
869379007	2020	9	17	0.033102	6efc93650669dc92ee31b13ba412a00dcfffb5cf13f2c...
925512001	2020	9	4	0.067780	43787c75a09352cf10839c1fa84a14d49a2fb8da93f7d1...
921266001	2020	9	1	0.016932	7a4379a7d887099f4134e1cce73f05e30285f3cbcc90fb...
922037003	2020	9	20	0.065525	69c9024fd15a772ae51e666c02a85b53d7bc7aead6e5a2...
728703002	2020	9	9	0.042356	eeefd55a289a090d4a1f85083554ed6302b47a51f87f2e...

100 rows x 5 columns

Figure 5 - sample dataset created from transactions dataset

Later the accuracy and performance of a KNN model will need to evaluate. Cross validation was used to resample the training and test dataset as the KNN model needed to be evaluated on a limited

sample of the data CV is specifically needed when we want to estimate how well a model performs on unseen data. Cross validation allows the result to be less biased than other methods such as a train/test split. However, our data consisted of a time series, and we were trying to forecast future product purchases by H&M customer. We cannot choose random samples, usually found in kFold cross validation. A Time Series Split was used. It created a simple split of 3 groups with the end of each group used for testing. Randomisation was not carried out ensuring the time dimension is kept intact. The X features consisted of the day of the month and the y predictor was customer ids.

```
In [41]: tscv = TimeSeriesSplit()
          print(tscv)
          TimeSeriesSplit(gap=0, max_train_size=None, n_splits=5, test_size=None)

In [42]: TimeSeriesSplit(max_train_size=None, n_splits=3)
Out[42]: TimeSeriesSplit(gap=0, max_train_size=None, n_splits=3, test_size=None)

In [48]: for train_index, test_index in tscv.split(X):
          #print('TRAIN:', train_index, 'TEST:', test_index)
          X_train, X_test = X[train_index], X[test_index]
          y_train, y_test = y[train_index], y[test_index]

In [179]: y_test[15]
Out[179]: 'eeefd55a289a090d4a1f85083554ed6302b47a51f87f2e400e151dde65fc9b71'
```

Figure 6 - Time Series cross validation training and testing split

Training the Model

We fit an unsupervised Nearest Neighbour model on normalised training data `X_Scaled` settling on `n=12` with a radius of 4, which means four of the closest neighbours have their distance computed. A test sample `q_scaled` was made on the first item in our test dataset with the settings to return 12 nearest neighbours of our test sample. This was to make a single prediction and see the result.

```
PRODUCT: 921748002
Oversized shirt jacket in cotton corduroy with a collar, buttons down the front and a yoke at the back. Flap chest pockets with a button and diagonal welt front pockets. Dropped shoulders, long sleeves with buttoned cuffs, and short slits in the sides. Slightly longer at the back. Unlined.

PRODUCT: 903004003
Trousers in a recycled polyester weave. High waist with pleats at the front, a zip fly with a hook-and-eye fastening, side pocket, fake back pockets and wide, straight legs with creases.

PRODUCT: 811899001
Fully lined swimsuit with a low-cut back and high-cut legs. Cups with removable inserts that shape the bust and provide good support.

PRODUCT: 920283001
Jumper in a soft, fine, fluffv knit with a square neckline front and back, long puff sleeves and ribbing at the cuffs and hem.
```

Figure 7 - four of 12 products identified by KNN

Evaluating the Model

50 transactions from a future month (September 2020) and 50 transactions from past month (August 2020) were taken and put into a collection called samples. Using cross validation Time Series split a `X_train`, `X_test` was created and used to train a KNN model to predict their future purchases. These were the days of month and price. `y_train` and `y_test` consisted of customer_ids. The idea here was based off model-based collaborative filtering assumption that customers who in the past bought similar products may buy similar products in the future. So, if we can find similar customers we can recommend products they bought in the past.


```

In [215]: precision_score(y_test, y_pred, average='micro')
Out[215]: 0.125

In [216]: accuracy_score(y_test, y_pred)
Out[216]: 0.125

```

Figure 8 - results from precision and accuracy score on sample transactions

Out of 16 actual and 16 predicted customers checked. The accuracy score result was a poor 12.5%. This was the ratio of correct predictions vs what was observed. This meant 2 customers were predicted correctly out of 16. Precision score looked at out of each sample fold checked 12.5% of items in the actual matched the predicted. Both accuracy and precision can be the same as it just means that every time, we checked different samples of our actual observation and our prediction they were similar. However, this could be future leakage of data into past data. Other techniques could be considered such as bootstrapping.


Generating Kaggle Submission File

To generate a submission file missing transaction by customers who did not purchase anything during data collection period, needed to be filled. Transactions were grouped by customers. The median day and price were recorded. It was found that out of 1,371,980 customers only 1,362,281 customers made transactions. A new dataset was created that consisted of customer_id, day and price. Both transaction customers and non-transaction customers were merged. There was 9,694 missing days and prices. The median was used to fill these gaps. A python function was created that took in the customer dataset split it into training and testing datasets, it opened a CSV file, wrote headers, and then looped through the customer dataset. The day and price for each customer was parsed into an unsupervised nearest neighbour model returning 12 neighbours. These similar customers were then used to search the transaction dataset of past purchases. When a product was found, it was then added to the recommended list (no popularity score was given). Each customer and 12 recommended products were then inserted into a CSV file. After the submission file was created, it was checked and then submitted to Kaggle for evaluation.

Kaggle gave the model a score of 0.0004 MAPS@12 (Mean average precision). This score is more suited to image classification, according to (Chanda, 2022) was the submission was poor when compared to the top score of 0.0364. It seems day of month and price were not the only factors to consider when predicting customer purchasing behaviour and when recommending products to customers. From reviewing leader board discussions processing image boosted scores.

Leaderboard [Raw Data](#) [Refresh](#)

YOUR RECENT SUBMISSION



ros_predictions3.csv
Submitted by Rob O'Sullivan · Submitted a few seconds ago

Score: 0.0004
Private score:

[Jump to your leaderboard position](#)

Search leaderboard

Public Private

This leaderboard is calculated with approximately 1% of the test data. The final results will be based on the other 99%, so the final standings may be different.

Prize Contenders



#	Team	Members	Score	Entries	Last	Code
1	senkin13		 0.0364	86	5h	

Figure 9 - Kaggle Submission Evaluation Score

Conclusion

H&M wanted a product recommender. A review of research in the field of fashion recommendation was conducted. This highlighted that time of month, social membership, clothing size, colour, brand, visual features in images, texture style and textural features are all important features for user and item representation. Also, that K nearest neighbour was a starting point for recommendations.

From an EDA it was found that H&M's dataset to process was large and unlabelled. An unsupervised nearest neighbour model was chosen to cluster customers. days of the month and price were chosen as features because they required no data cleaning and KNN algorithms do not perform well on many features. The transaction dataset was prepared and split into three groups using cross validation time split. Training data was then scaled and passed into a Nearest Neighbour model using $k=4$. The model was asked to return 12 closest customers. Products these customers purchased in the past were then recommended to customers who were queried. This querying was carried out on every customer in the customers dataset. A csv file was generated and submitted to Kaggle.

Local precision and accuracy score for a Nearest Neighbour model was 12.5% when 16 predicted customers were compared to 16 actual customers. H&M used MAPS@12 to evaluate the csv submission file. H&M gave the file a score of 0.0004 (leader was 0.0364).

Feature Selection Opportunities

Nearest Neighbour was used because we had an unlabelled dataset and day-price was used due to its data cleaning convenience. Chi-squared testing and Fisher's score could have been used to identify what features impacted the customer classification. This could have been packaged into a Recursive Feature Elimination to choose the best features for prediction.

Dimension Reduction Opportunities

A Principal Component Analysis (PCA) (Galarnyk, 2015) or Single Value Decomposition (SVD) (Kumar, 2020) could be used to reduce the number of dimensions used in the prediction. SVD is suitable for sparsity in datasets and a variation of this was used to by Simon Funk to win the Netflix recommender competition on Kaggle.

Memory Management Opportunities

Utilising garbage collection, incorporating SQLite or a database system, using GPU for processing via TensorFlow or PyTorch, could reduce memory and CPU usage related to analysing, cleaning, and preparing the data for machine learning.

Model Opportunities

The entire transaction data set was merged with the customer and articles dataset, then Multi KNN Classifier was explored. Accuracy and precision were improved when this dataset was sampled. However, scaling this to a submission file was not possible so it was abandoned for a simpler solution. An ensemble may possible using KMeans clustering to generate labels that could be fed into a Multi KNN classifier. This pipeline should be built next for fashion recommendation and may see improved results. Computer vision is important and convolutional neural networks should also be considered on the 30gb image dataset to create features that could be fed into the ensemble.

References

- Bellini et al. (2022, January 13). *Multi Clustering Recommendation System for Fashion Retail*. Retrieved from Multimedia Tools and Applications: <https://doi.org/10.1007/s11042-021-11837-5>
- Brownlee, J. (2020, May 11). *Singular Value Decomposition for Dimensionality Reduction in Python*. Retrieved from Machine Learning Mastery: <https://machinelearningmastery.com/singular-value-decomposition-for-dimensionality-reduction-in-python/>
- Chanda, D. (2022, February). *Understanding Mean Average Precision*. Retrieved from Kaggle: <https://www.kaggle.com/code/debarshichanda/understanding-mean-average-precision>
- Deldjoo et al. (2022, February 6). *A Review of Modern Fashion Recommender Systems*. Retrieved from Arxiv, Cornell University: <https://arxiv.org/abs/2202.02757>
- Ding et al. (2022, March 07). *Modeling Field-level Factor Interactions for Fashion Recommendation*. Retrieved from Arxiv, Cornell University: <https://arxiv.org/abs/2203.03091>
- Galarnyk, M. (2015, December 5). *PCA using Python (scikit-learn)*. Retrieved from Towards Data Science: <https://towardsdatascience.com/pca-using-python-scikit-learn-e653f8989e60>
- Jankiewicz, P. (2022, February). *HM - Create dataset samples*. Retrieved from Kaggle: <https://www.kaggle.com/code/paweljankiewicz/hm-create-dataset-samples>
- Karpov, D. (2022, March). *H&M EDA First Look*. Retrieved from Kaggle: <https://www.kaggle.com/code/vanguard/h-m-eda-first-look>
- Kelleher et al. (2020). *Fundamentals of Machine Learning for Predictive Data Analytics*. London: The MIT Press.
- Ko et al. (2022, January 03). *A Survey of Recommendation Systems: Recommendation Models, Techniques, and Application Fields*. Retrieved from Multidisciplinary Digital Publishing (MDPI): <https://www.mdpi.com/2079-9292/11/1/141>
- Kumar, V. (2020, March 25). *Singular Value Decomposition (SVD) & Its Application In Recommender System*. Retrieved from Analytics India Mag: <https://analyticsindiamag.com/singular-value-decomposition-svd-application-recommender-system/>
- Léa et al. (2018, May). *Influencers impact on decisionmaking among generation Y and Z*. Retrieved from Digitala Vetenskapliga Arkivet: <https://www.diva-portal.org/smash/get/diva2:1214227/FULLTEXT01.pdf>
- Licht Lab. (2022, March). *H&M data Deep Dive / chap.1 Understand article*. Retrieved from Kaggle: <https://www.kaggle.com/code/lichtlab/h-m-data-deep-dive-chap-1-understand-article>
- Mohammadi et al. (2021, November 01). *Single-Item Fashion Recommender: Towards Cross-Domain Recommendations*. Retrieved from Arxiv, Cornell University: <https://arxiv.org/abs/2111.00758>
- Sheth, V. (2020, November 05). *MultiClass Classification Using K-Nearest Neighbours*. Retrieved from Towards Data Science: <https://towardsdatascience.com/multiclass-classification-using-k-nearest-neighbours-ca5281a9ef76>

Singh, M. (2019). *KNN Multi Classification - Animal Classification*. Retrieved from kaggle.com:
<https://www.kaggle.com/code/martandsay/knn-multi-classification-animal-classification/notebook>

Souames, M. A. (2022, February). *Tips To Work Efficiently with the Dataset*. Retrieved from Kaggle:
<https://www.kaggle.com/code/souamesannis/tips-to-work-efficiently-with-the-dataset>

Zhu, Z., & Van Roy, N. (2021, September 26). *Deep Exploration for Recommendation Systems*. Retrieved from Arxiv, Cornell University: <https://arxiv.org/abs/2109.12509>

Appendix

Equations

$$\forall u \in \mathcal{U}, i_u^* = \operatorname{argmax}_{i \in I \setminus I_u^+} g(u, i)$$

Where:

\mathcal{U} = users (of which user u is a subset of \mathcal{U})

\mathcal{I} = items (of which item i is a subset of \mathcal{I})

i_u^* = best matching item not consumed by user yet
 (either via direct 1-5 like feedback or implicit feedback
 like, clicked on, added to cart, purchased before.

S_{ui} = user preference for item which is a subset of S .

I_u^+ = user-item pairs for each S_{ui} .

Equation 1 – Utility Function

$$\hat{s}_{u,i} = p_u^T q_i$$

Where:

- p_u, q_i = embedding vectors
for a user u and item i .

Equation 2 - Bayesian Personalised Ranking from implicit feedback (BPR-MF)

$$O^* = \operatorname{argmax}_{O_j \in \mathcal{O}} s(O_j)$$

Where:

s = outfit utility function score

Equation 3 - Fashion Outfit Composition Score

$$\hat{x}_{u,i} = p_u^T q_i + \theta_u^T \underbrace{E \Phi_f(\mathbf{Img}_i)}_{f_i}$$

Equation 4 - Visual Bayesian Personalised Ranking (VBPR)

$$\hat{x}_{u,i} = p_u^T q_i + p_u^T (E f_i - l_i)$$

Equation 5 - DeepStyle formula

Figures

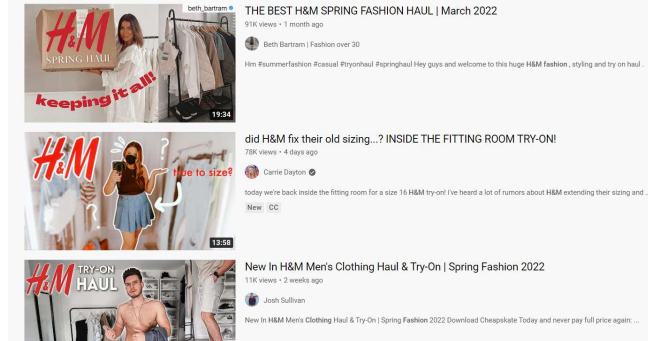


Figure 10 - Social media influences recommending products by time of year

Jupyter Notebooks

- ca2e2 – Nearest Neighbour Clustering Model
- ca2l – K-Nearest Neighbour Multi Class Classifier Model