Combinational Collaborative Filtering: An Approach For Personalised, Contextually Relevant Product Recommendation Baskets

Research Project - Jai Chopra (338

Dr Wei Wang (Supervisor)

Dr Yifang Sun (Assessor)

Introduction

- Recommendation engines are now heavily used online
- **35%** of Amazon purchases are from algorithms
- We would like to extend on current implementations and provide some mor erating **goal-oriented** item sets that are **complementa** mbinational item set recommendations)

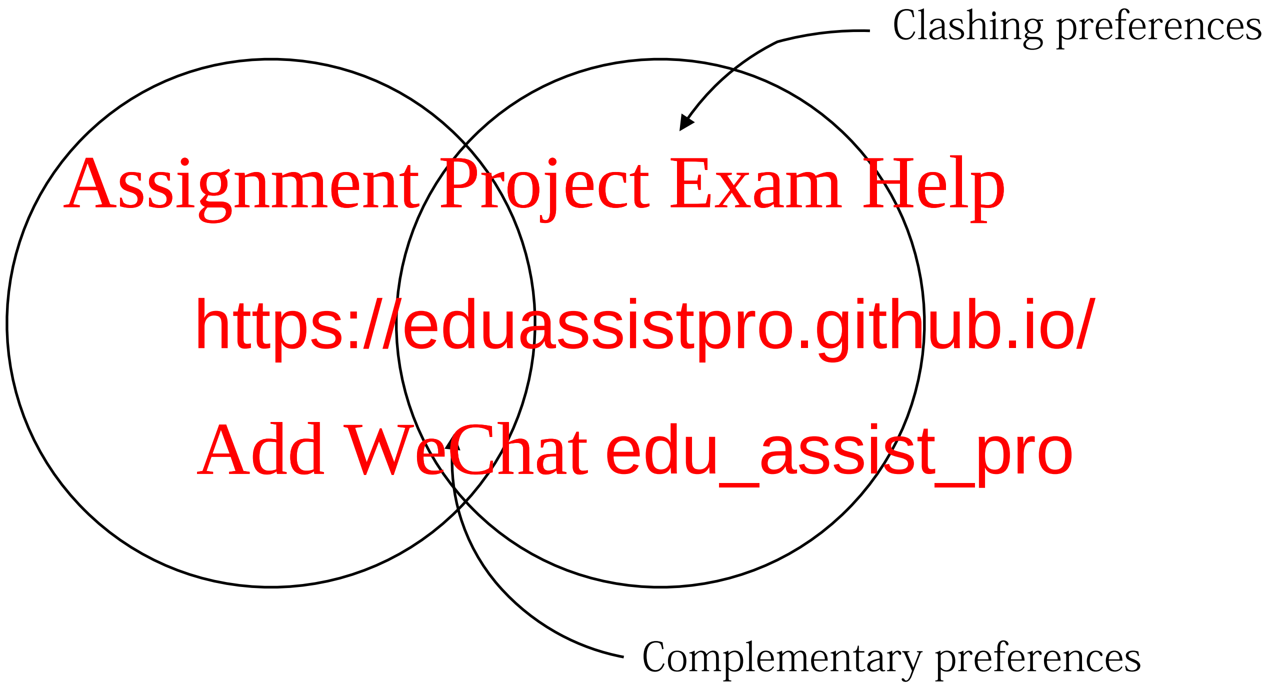
Company background

- Kent and Lime (KAL) is an online styling service
- Data driven business which collects style profile information, feedback and purchase history
- We would like to L dataset, and domain knowledge to produce an auton ling agent

Problem space definition

- Goal oriented recommendations
- Contextual recommendations
- Domain knowledge is highly relevant for the design of our system

Problem space definition



Implementation goals

- Recommendations delivered in a timely manner
- Complementary by nature
- Well suited to the
- Learn and perfor me
- Reasonable performance at the avoid cold start)
- Deployed online in a web application environment

Overview of presentation

- Data Preprocessing
- Recommendation engine implementation
- Web application d tecture discussion
- Demonstration
- Experiments and evaluation
- Future considerations

Data preprocessing - version mismatches

- Over time, profile schemas changed
- Solution: pick a subset of data that was common across all schemas

Data preprocessing - missing values

- Many missing values
- Solution: use a mean average, or an initialised value, or discard row

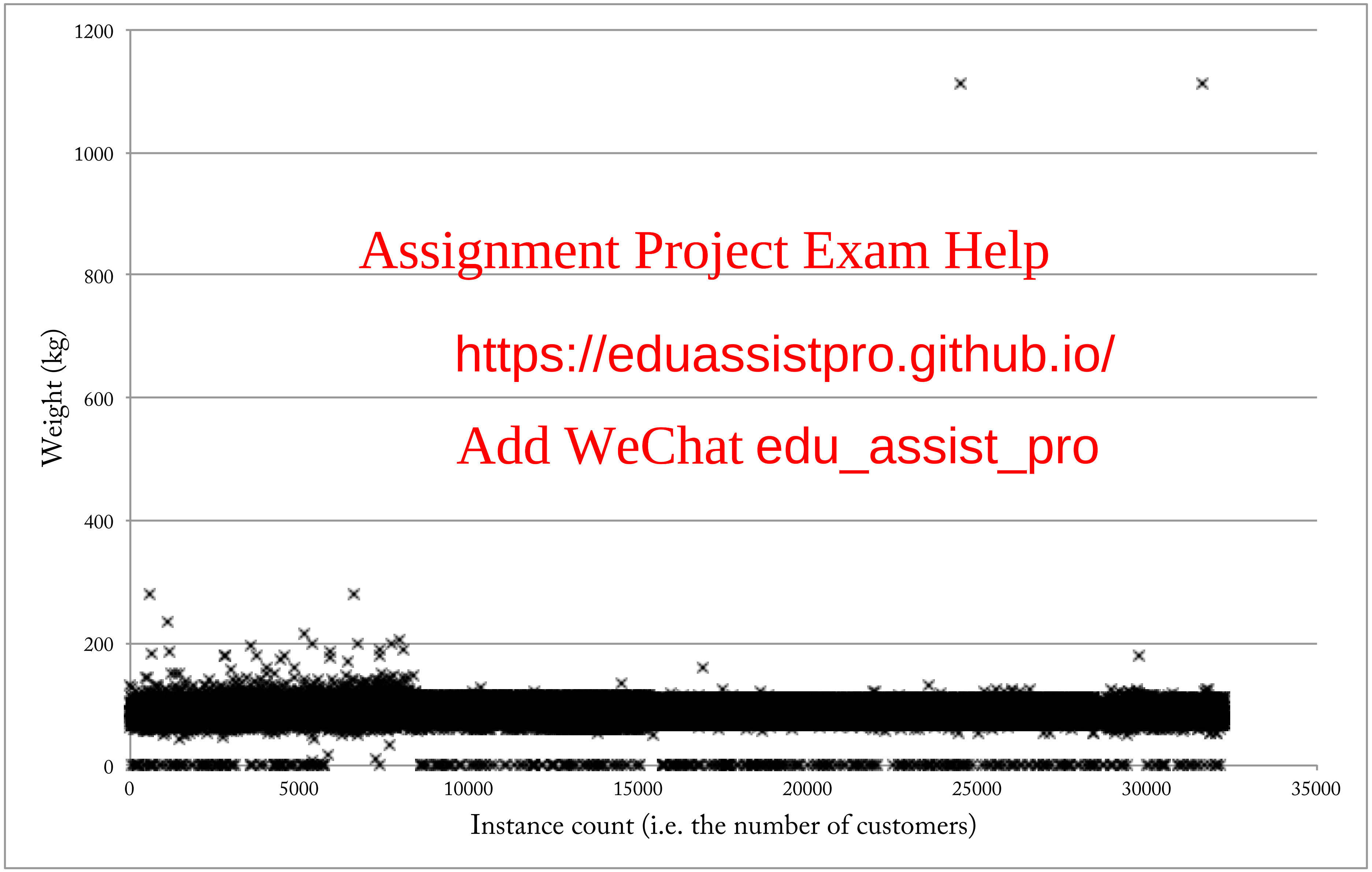
Data preprocessing - inconsistent fields

- Inconsistent values when merging different versions of the schema
- Solution: pick a co ansform

Data preprocessing - outliers

- Outliers were discovered after the implementation had started, producing highly skewed results
- Retrospectively ha ved) after some analysis

Data preprocessing - outliers



Data preprocessing - initialised state objects

- Define an initialised state for each customer row

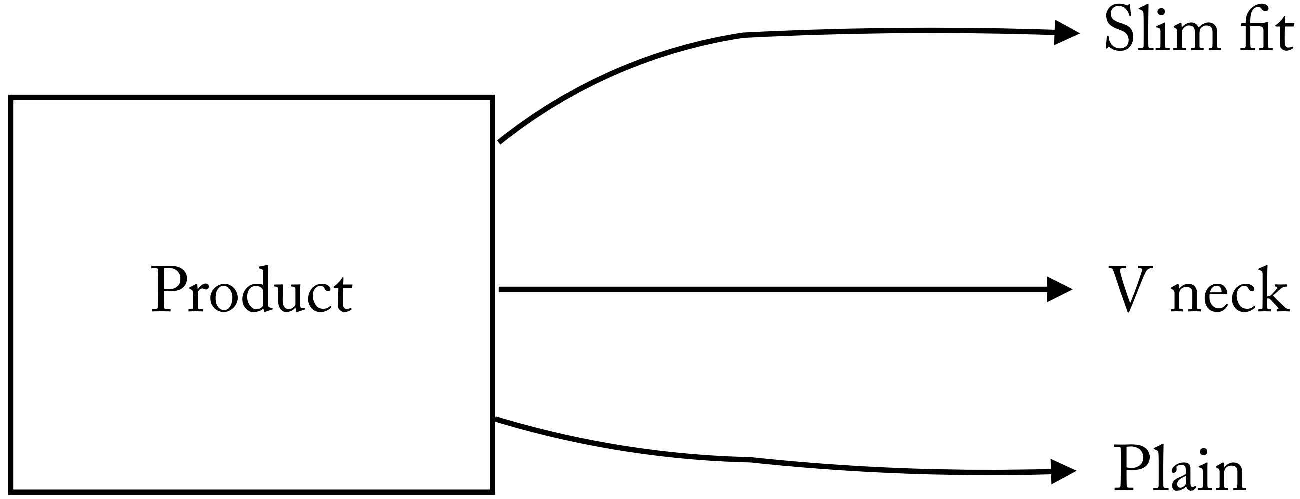
Initialise state: One-hot encoding:

[0,0,0 1,0]

- Run a check to see, *how dissimilar the row is to the initialised state*. If dissimilarity < threshold, clean/remove row

Data preprocessing - product content data

- Tags are metadata which allow for item to item filtering and some pre-selection
- However, we are se on *older* products that do not have tags, t



Implementation (Recommendation Engine)

- Xue et al describes two broad types of CF:
  - Memory based approaches
  - Model based app
- There are also hyb ystem would like to use both techniques, primarily me
- Pennock et al describes a hybrid approach using *'personality type'*, where customers have some pre-selection based on what personality they are

Content boosting using product tags

- Problem: CF will only suggest products that have been purchased previously (bias toward *older* products)
- KAL dataset cont this to perform some clustering based o
- Classification solves:
  - Newer products not being selected
  - Evaluation techniques, as it is too difficult to classify on a granular product based level
  - Products are usually out of stock (OOS)

Product clustering design decisions

- What value of K?
  - 5 was selected after analysing results with inventory staff

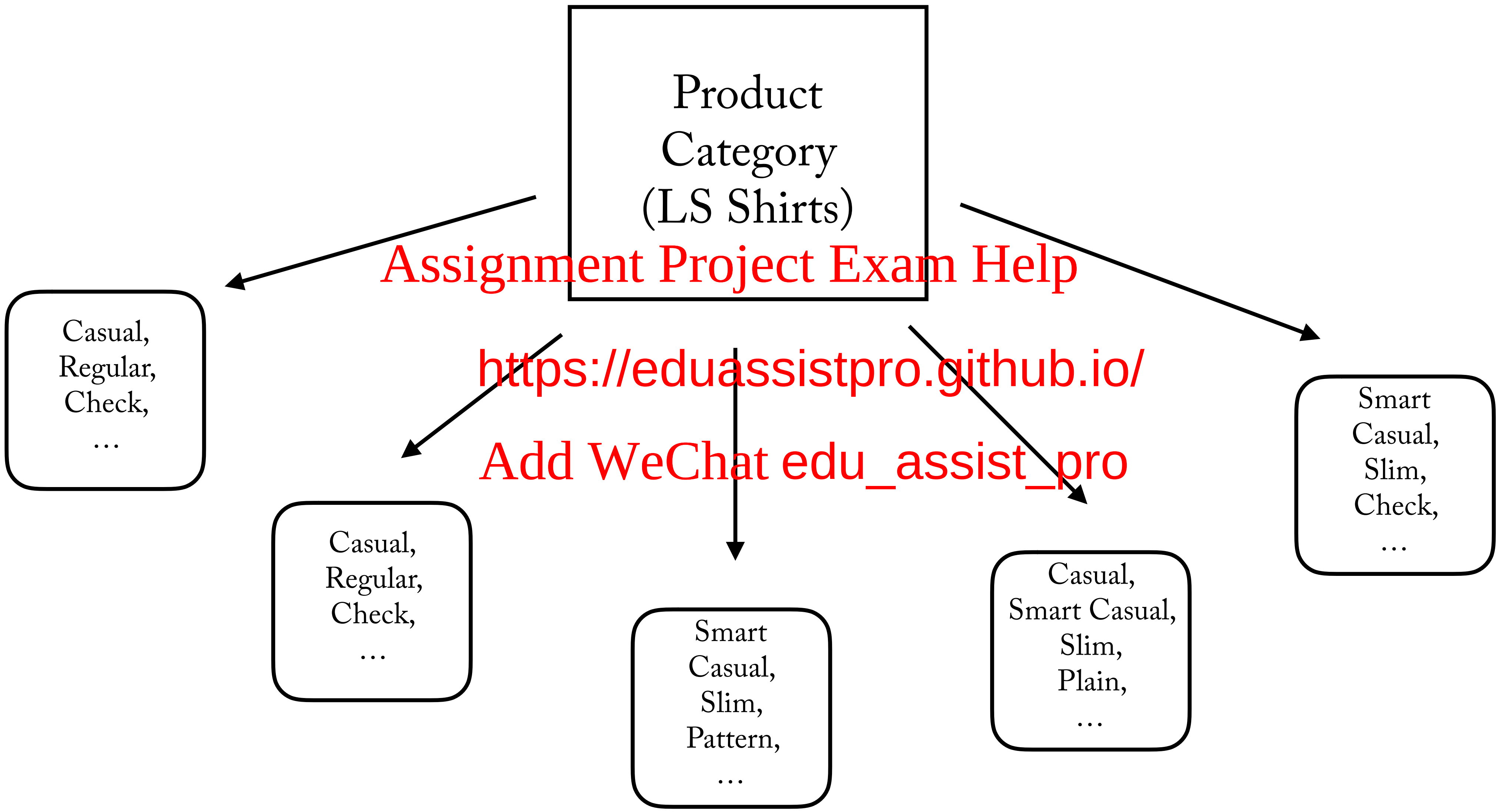

```
Clustered Instances

 0       78 ( 22%)
 1      113 ( 32%)
 2       53 ( 15%)
 3       37 ( 11%)
 4       71 ( 20%)
```

Product clustering design decisions

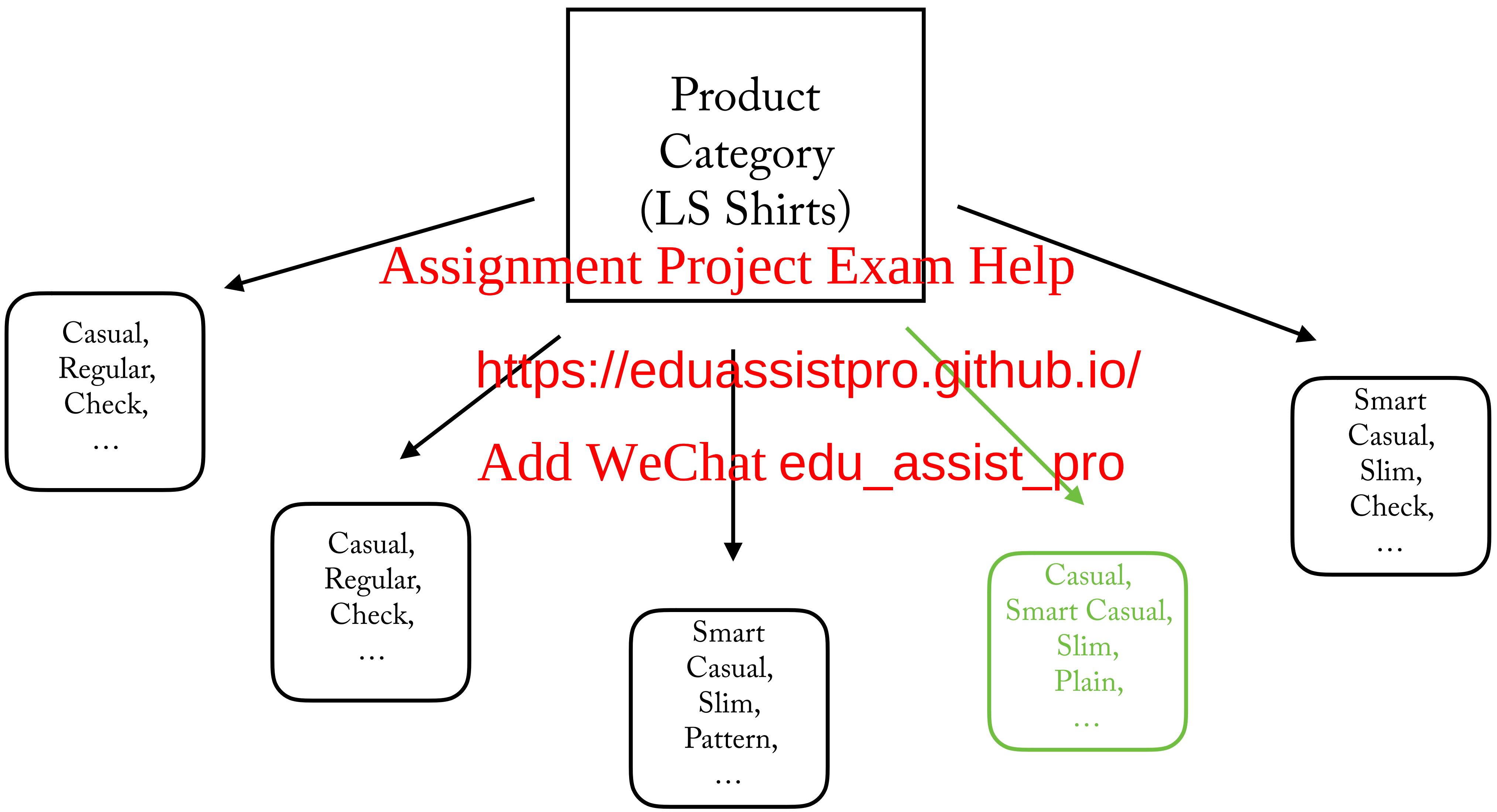
```
Final cluster centroids:
                                        Cluster#
Attribute                  Full Data          0          1          2          3          4
                             (352.0)     (78.0)    (113.0)     (53.0)     (37.0)     (71.0)
=========================================================================================
Casual10                           1          1          0          1          0          1
SmartCasual11           0
Dress12      0          0
Regular13                          1          1                     0          0          1
Slim14                             0          0                     1          1          0
Check15                            0          1          0          0          1          1
Pattern16                          0          0          1          0          0          0
Stripe17                           0          0          0          0          0          0
Plain18                            0          0          0          1          0          0
```


Cluster 0: {Casual, Regular, Check}

Goal is to select product class, not product



Goal is to select product class, not product



How do we determine winning class?

- Problem: We wish to find the winning class
- Naively, we can find the winning product and abstract this to its class, but this is
- Thus, we wish to for each class, and pick the highest one

Data: List of product categories matrices P, with length l
Result: Highest scoring product category (or class)

$p_{max} \leftarrow 0$;
p_{best};
for $i \leftarrow 0$ **to** l **do**
    $p_{score} \leftarrow$ averageProductScore($P[i]$);
    if $p_{max} < p_{score}$ **then**
        $p_{max} \leftarrow p_{score}$;
        $p_{best} \leftarrow P[i]$;
    end
end

Algorithm 1: Determine the winning product classification from the average product score in each cluster

Problems with selecting winning class

- Dominant product scores
  - Products that are older have more rating (time is biased)
  - Products that ne

- Sparsity
  - No rating products can thus i he class selection

Thresholding dominant product scores

- Use a generalised logistic function (modified)

Smoothing for “unrated” products

- Common technique used to reduce sparsity
- Select the average score of the cluster and associate with unrated products

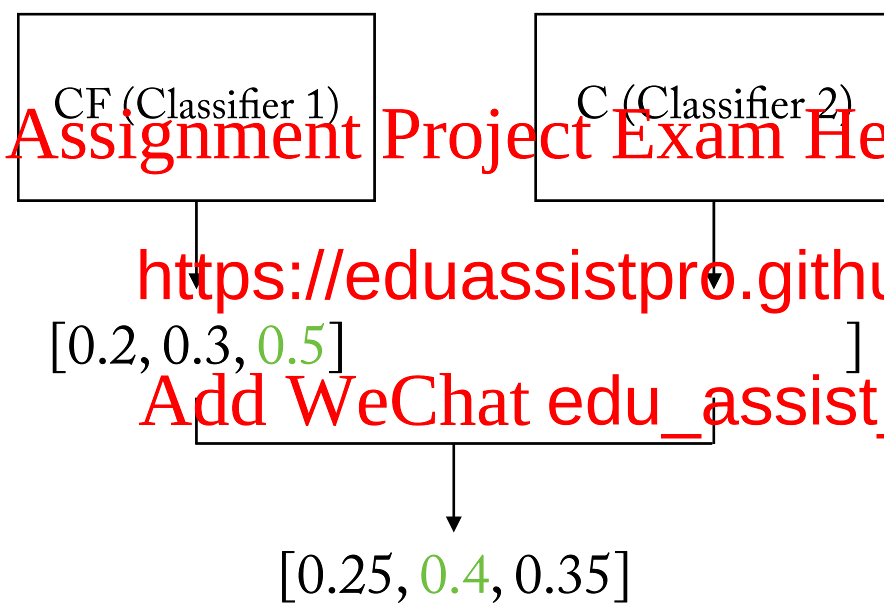
Noise reduction

- Remove customers that have not purchased more than 2 times (i.e. 2 baskets)

“Slow start” problem of CF

- Customers that have never purchased before have poor system accuracy
- How do we resolv a model where similarity is built on custom an purchase history
- We can then combine “votes” f t classifications in an ensemble classifier

“Slow start” problem of CF



“Slow start” problem of CF

- Introduce a weighted “voter” function, and treat each category classification as a probability. Similar to MLE.

Self weighting

- We also want to vote the *active* users votes higher than its peers in the neighbourhood, if it has rated products or given feedback

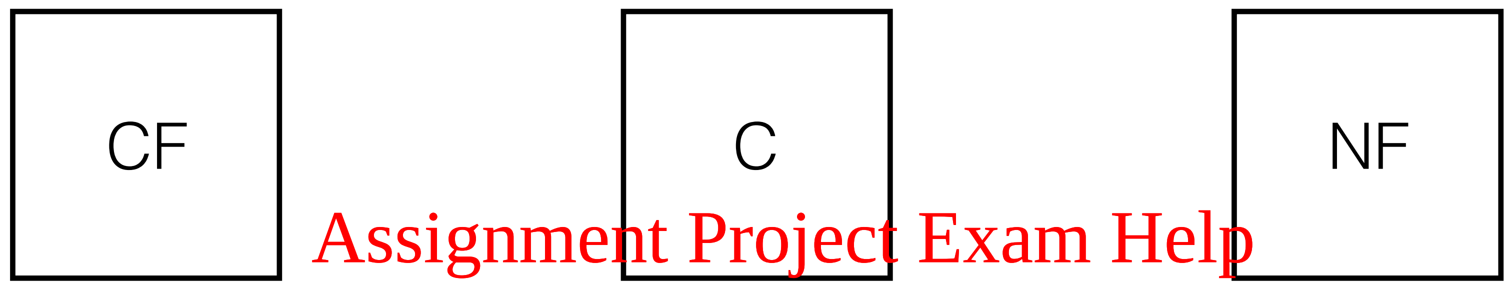
Design decisions - producing customer clusters

- Initially k-means was unable to produce some nice clusters that reflected both our fit and style knowledge domains
- We prefer to weig higher over non-physical attributes ists *naturally* does)
- We also *increase k* to capture bo owledge domains

Feedback data incorporation

- KAL dataset contains negative explicit feedback based on style, as well as negative implicit feedback based on fit
- Author chose to d ... ack (too noisy/ uncertain)
- Explicit feedback was used to *r ... t classifications*, and then subtract a value from the final vote, i.e. add to the ensemble method

Ensemble voting classifier



Output votes for p ass

Select the highest product class

Building combinational (complementary) baskets

- Use rule mining to build IF-THEN statements so we can modify our baskets before our final item sets
- Look for common uct classifications e.g. If

`ckt_S_3hirts_1,`

Web architecture design decisions

- The goal of the project is to implement an efficient recommendation system, thus the web application itself should be *fast*

Node streams and HTTP chunks

- Use data frames to split a product list into products
- Operate on each product, rather than a data frame
- Send the product t next data frame is processing

Optimising database queries

- Filter noise (i.e. only query customers with purchases > 2)
- Filter OOS products (useful for item to item filtering)
- MongoDB Pipelin

Online / incremental updates

- Once a customer has given feedback, we wish to suggest them a new item immediately
- Thus, we want our at is, it updates incrementally
- With instance based learning te his is possible

Experiments and evaluation

- Experiments:
  - Does our ensemble method increase accuracy? What value of lamda do we use?
  - Optimisation me... of weighting, reducing noise impact
  - Speed of transfer in web

Experiments and evaluation

- Measure to use, mean absolute error (MAE)

Experiments and evaluation

Experiments and evaluation

Experiments and evaluation

Production results

- Deployed to production on 18 November

Future work and considerations

- Implement new rules based on stylist domain knowledge with respect to product attributes: colour, seasonality, time-variant features
- Filter categorical pr ant seasons
- Testing memory-based CF vs d CF (i.e. given training set, build model, probabilistic measure of likely product classification)
- Explore other research based methods, genetic algorithms

Conclusion

- Very interesting project
- Many different fields of CS - Machine learning, data mining, data warehousing, lopment, human centered design
- Ground work for future potenti rea
- Thank you for listening