Product Search Relevance

Rohan Sadale, Akshay Kulkarni, Gautham Sunder and Utkarsh Kajaria

Data

- 240k (query,product) tuples
- 24k unique queries
- 97k unique products
- Product info:
  - Title
  - Description
  - Attributes
  - Relevance:
    - A continuous number between 1 and 3.
    - 1 - lowest, 3 - highest

Search Term	Product Title	Product Description	Product Attributes	**Relevance**

Problem Description

Given a search term, can we predict the relevance score of product?

Approach

- Classification
  - Two-class
  - Three-class
- Regression

Text Preprocessing

- Replace symbols and patterns
- Remove/replace words in parentheses with dots/space
- Unit Converter
  - pounds | pound | lbs | lb | lb. => “lb”
  - wattage | watts | watt => “watt.”
- Remove commas between digits
  - 10, 000 will be replaced with 10000.
- Number to Digit Mapper
  - “one” => 1; “two” => 2.

```
def str_stem(s):
    s = s.lower()
    s = s.replace(",","")
    s = re.sub('(?<=[0-9])[\ ]*centimeter[s]*(?=\ |$|\.)', '-cm ', s)
    s = s.replace("vynal","vinyl")
    s = stemmer.stem(s)
    return s
```

Feature Engineering

- Generated 26 features
- A combination of
  - basic descriptive features
  - distance features
  - word2vec embedding features
  - intersect count
  - position features
- Distance between search term and product title/product description
  - Jaccard Distance
  - Edit Distance

Feature Engineering

- Intersection based features
  - First and Last
  - Intersect count
  - Intersect position
- Word2Vec and Doc2Vec
  - Average Similarity
  - Centroid RMSE
- Longest Match
- TF-IDF Scores
- Asymmetric features that are non contextual such as presence of product dimensions

Models

- Linear
  - NaiveBayes
  - Logistic Regression
- Ensemble
  - Adaboost
  - Random Forests
- Neural Networks
- Regression

Experimental Setup

- Parameter Tuning
  - Logistic Regression   lambda = [0.001, 0.01, 0.1, 0, 1, 10, 1000]

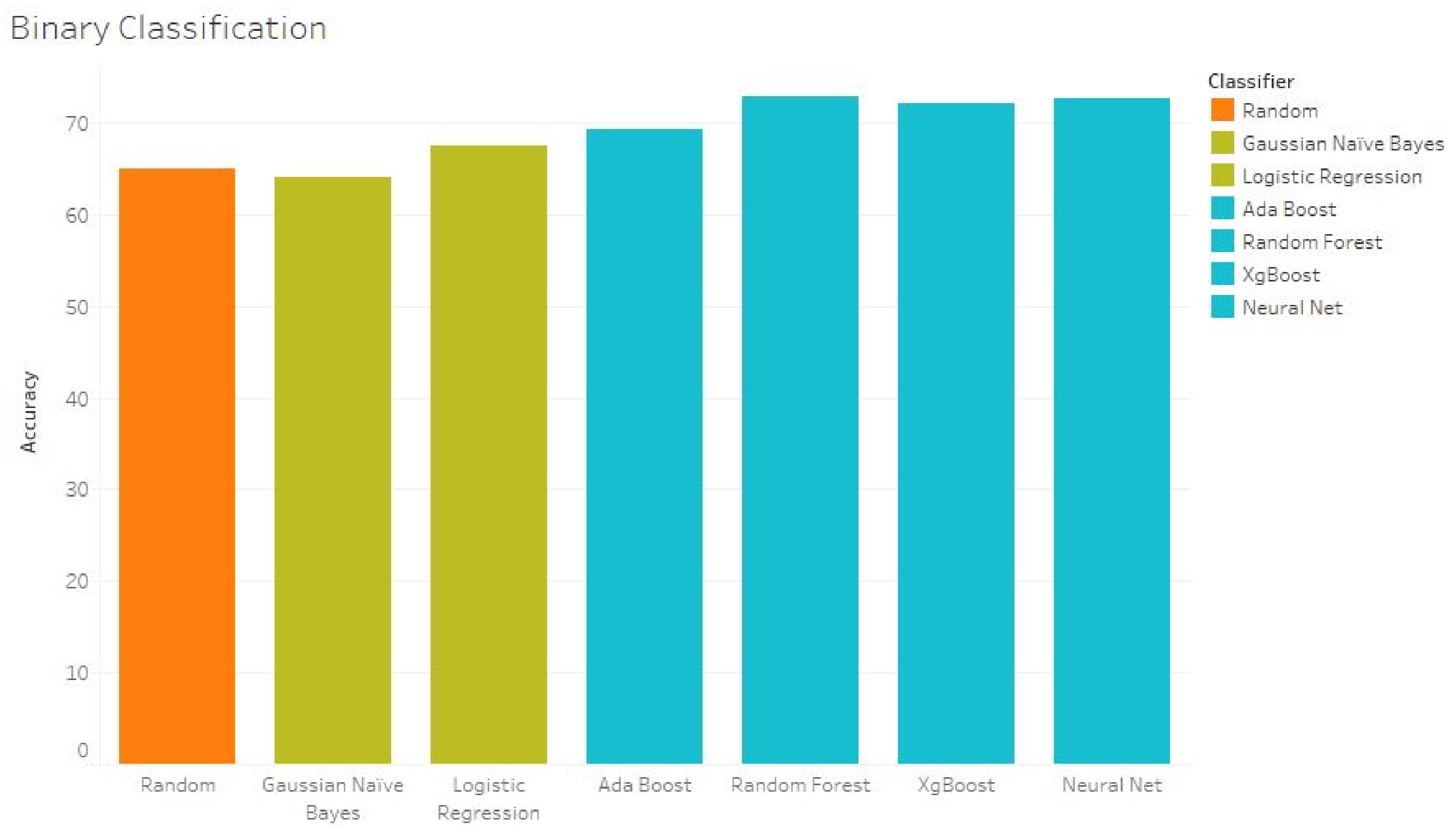
  - Random Forests
    - num_estimators [300, 500]
    - criterion : [‘gini’, ‘entropy’]
    - Max_features : [‘auto’, ‘log2’, None]

  - AdaBoost
    - criterion : [‘gini’, ‘entropy’]
    - max_features : [‘auto’, ‘log2’, None]

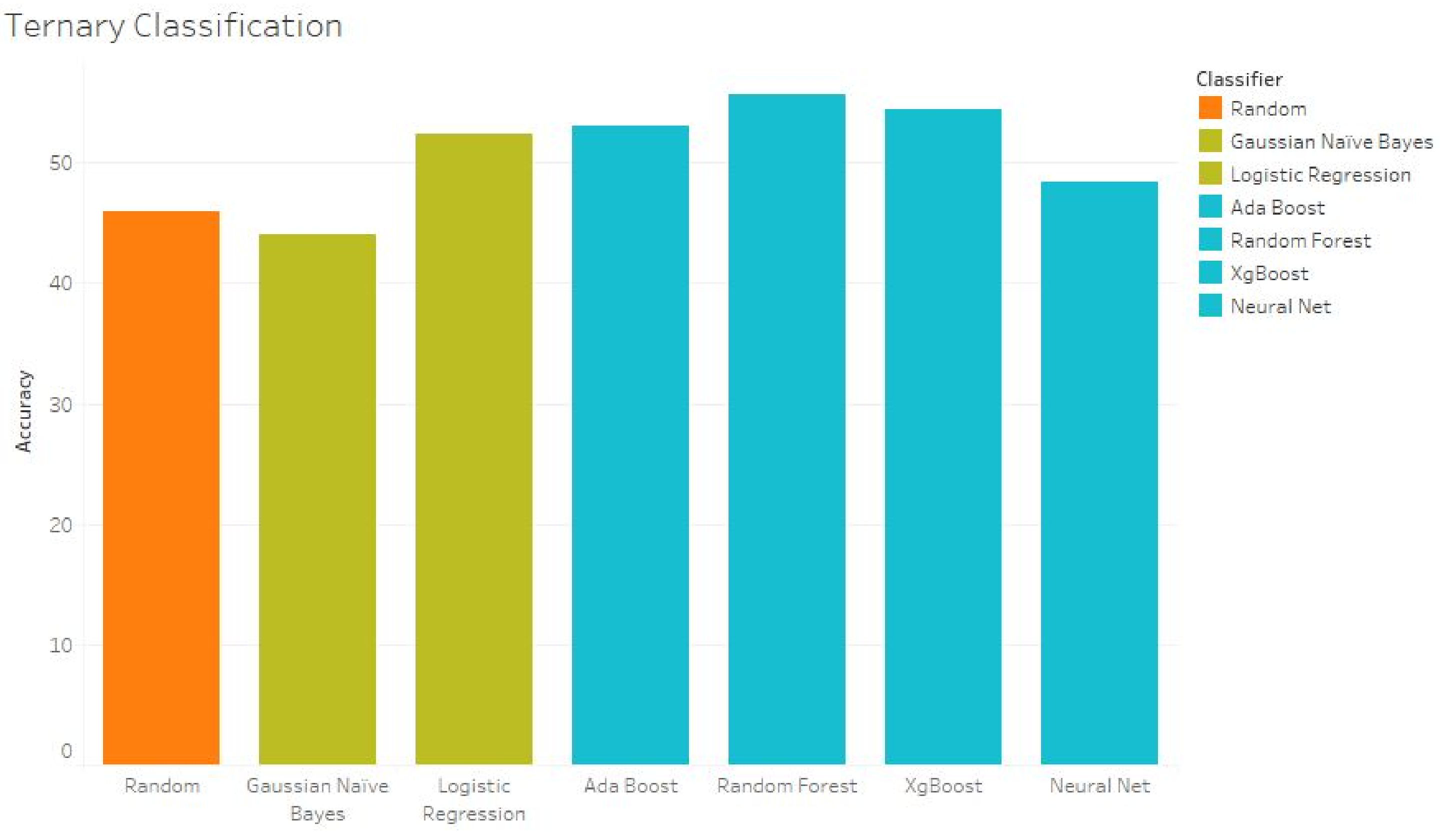
Experimental Setup

- Parameter Tuning
  - Neural Network
    - Num_hidden_layers : [1,2,3]
    - Activation : [‘relu’,’sigmoid’]
  - Regression - XGBoostRegressor
    - n_estimators : [100,150,200]
    - Max_depth : [3,5,7,9]
- Model Evaluation
  - Cross Validation
  - F - measure

Results (Two-class classification)



Results (Three-class classification)



Results (Regression)

XgBoost Regression yielded an RMSE 0.47636 on the test data

Finished in the top 25% of Kaggle leaderboard (~2100 participants)

Thank you

Questions?