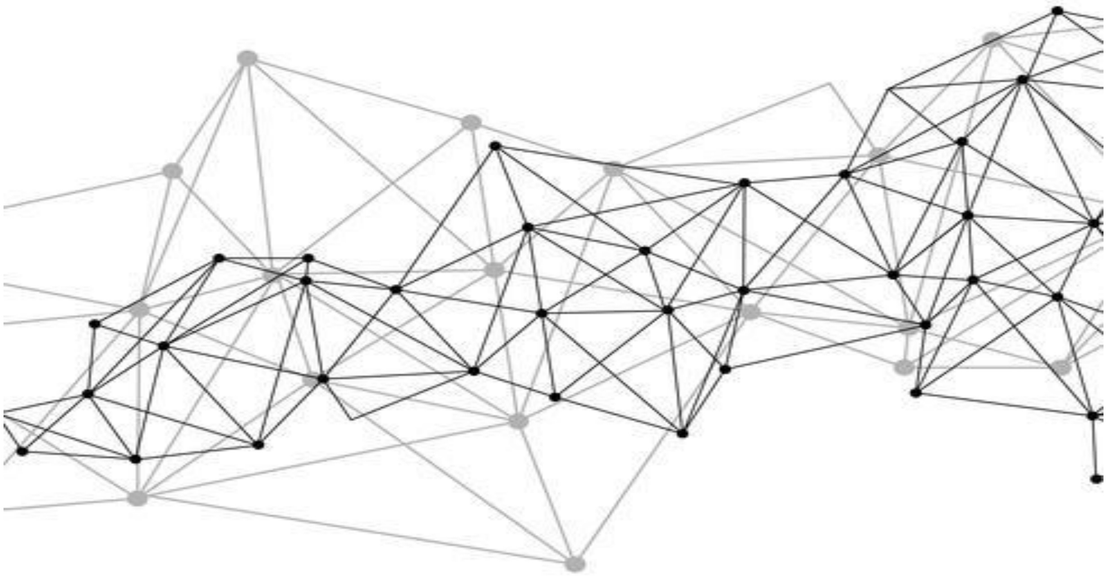
OPTIMIZATION FOR DATA SCIENCE

PREDICTION OF WOMEN'S BEST SELLING CLOTHES BASED ON E-COMMERCE REVIEWS



Submitted by

Priyanka Kumar

MSc in Data Science & Analytics (EPITA – 2020)

INTRODUCTION

In today's business world, customer reviews of a product in the E-commerce platform is of prime importance for defining marketing strategies of organizations. Companies are focused on analyzing the behaviour of the customers to derive insights because they (a) influence buying decisions (b) build trust and credibility (c) genuinely make business better (d) boost Search Engine Optimization(SEO) (e) are a loyal lot. Besides, it will also give them an idea on how to improve their offers and retain their customers. This project intends to predict Women's best-selling clothes based on customer reviews provided an E-commerce platform. To achieve this target, we employed the gradient descent algorithms on dataset features, and we implemented optimization techniques to improve the performance of the model.

DATASET

The dataset which has been used in this project is **Women's Clothing E-Commerce Reviews** dataset revolving around the reviews written by customers. Its supportive features offer a great environment to parse out the text through its multiple dimensions. Because this is real commercial data, it has been anonymized, and references to the company in the review text and body have been replaced with "retailer".

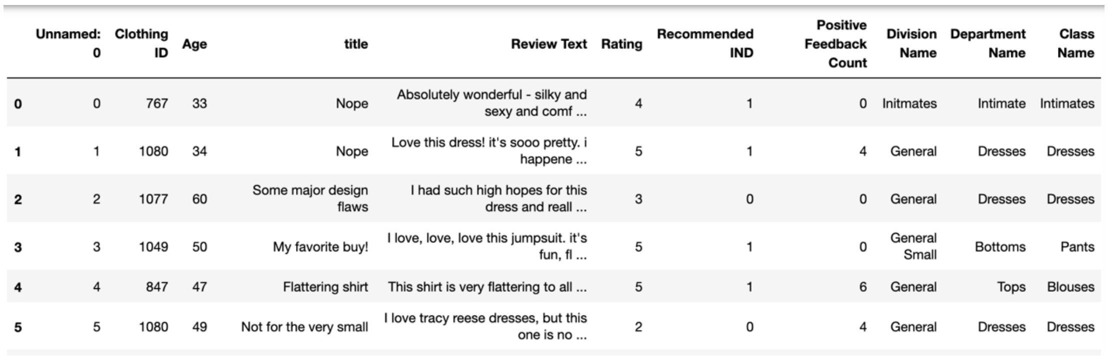
METADATA

There are 10 features in the dataset, each row corresponds to a customer review, and includes the following variables

Column Name	Description
Clothing ID	Integer Categorical variable that refers to the specific piece being reviewed.
Age	Positive Integer variable of the reviewers age
Title	String variable for the title of the review.
Review Text	String variable for the review body
Rating	Positive Ordinal Integer variable for the product score granted by the customer from 1 Worst, to 5 Best.
Recommended IND	Binary variable stating where the customer recommends the product where 1 is recommended, 0 is not recommended.
Positive Feedback Count	Positive Integer documenting the number of other customers who found this review positive.

Division Name	Categorical name of the product high level division.
Department Name	Categorical name of the product department name.
Class Name	Categorical name of the product class name

PREVIEW



	Unnamed: 0	Clothing ID	Age	title	Review Text	Rating	Recommended IND	Positive Feedback Count	Division Name	Department Name	Class Name
0	0	767	33	Nope	Absolutely wonderful - silky and sexy and comf ...	4	1	0	Initmates	Intimate	Intimates
1	1	1080	34	Nope	Love this dress! it's sooo pretty. i happene ...	5	1	4	General	Dresses	Dresses
2	2	1077	60	Some major design flaws	I had such high hopes for this dress and reall ...	3	0	0	General	Dresses	Dresses
3	3	1049	50	My favorite buy!	I love, love, love this jumpsuit. it's fun, fl ...	5	1	0	General Small	Bottoms	Pants
4	4	847	47	Flattering shirt	This shirt is very flattering to all ...	5	1	6	General	Tops	Blouses
5	5	1080	49	Not for the very small	I love tracy reese dresses, but this one is no ...	2	0	4	General	Dresses	Dresses

METHODOLOGY

IMPLEMENTATION & EVALUATION OF ALGORITHMS

The object of this exercise is to study the experimental analysis of various optimization algorithms on the given dataset (Women's E-commerce Clothing Reviews Dataset). We are implementing the following algorithms, namely Random Forest Classifier, Stochastic Gradient Descent Classifier and Support Vector Classifier. As a final step, we have executed Cross Validation Score technique to improve the performance of Random Forest Classifier model.

1. RANDOM FOREST CLASSIFIER

Random forests are a supervised learning algorithm. It can be used both for classification and regression. It is also the most flexible and easy to use the algorithm. A forest is comprised of trees. It is said that the more trees it has, the more robust a forest is. Random forests create decision trees on randomly selected data samples, gets a prediction from each tree and selects the best solution by means of voting. It also provides a pretty good indicator of the feature importance.

Random forests have a variety of applications, such as recommendation engines, image classification and feature selection. It can be used to classify loyal loan applicants, identify fraudulent activity and predict diseases. It lies at the base of the Boruta algorithm, which selects important features in a dataset.

RANDOM FOREST CLASSIFIER

```
Entrance [215]: rfc  =  RandomForestClassifier ( n_estimators = 200 )
                rfc . fit ( X_train , Y_train )
                pred_rfc  =  rfc . predict ( X_test )
```

```
Entrance [216]: #This is how our model is performing now
                print ( classification_report ( Y_test , pred_rfc ))
```

```
                  precision recall f1-score support

               0 0.78 0.80 0.79 851
               1 0.95 0.95 0.95 3847

        accuracy 0.92 4698
       macro avg 0.87 0.87 0.87 4698
    weighted avg 0.92 0.92 0.92 4698
```

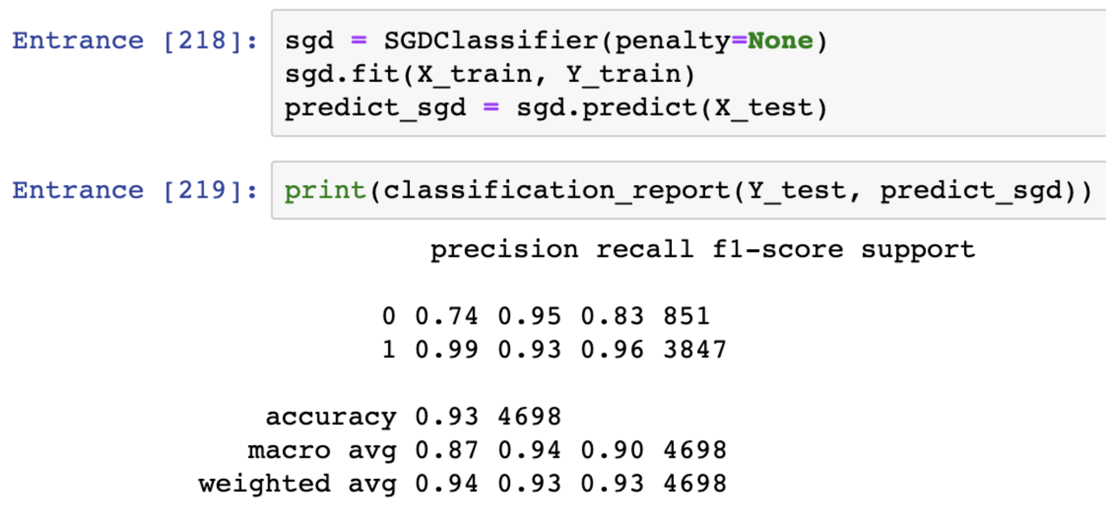
2. STOCHASTIC GRADIENT DESCENT CLASSIFIER

In Stochastic Gradient Descent, a few samples are selected randomly instead of the whole data set for each iteration. In Gradient Descent, there is a term called “batch” which denotes the total number of samples from a dataset that is used for calculating the gradient for each iteration. In typical Gradient Descent optimization, like Batch Gradient Descent, the batch is taken to be the whole dataset. Although, using the whole dataset is really useful for getting to the minima in a less noisy or less random manner, but the problem arises when our datasets get really huge.

Suppose, we have a million samples in our dataset, so if we use a typical Gradient Descent optimization technique, we will have to use all of the one million samples for completing one iteration while performing the Gradient Descent, and it has to be done for every iteration until the minima are reached. Hence, it becomes computationally very expensive to perform.

This problem is solved by Stochastic Gradient Descent. In SGD, it uses only a single sample, i.e., a batch size of one, to perform each iteration. The sample is randomly shuffled and selected for performing the iteration. So, in SGD, we find out the gradient of the cost function of a single example at each iteration instead of the sum of the gradient of the cost function of all the examples. In SGD, since only one sample from the dataset is chosen at random for each iteration, the path taken by the algorithm to reach the minima is usually noisier than your typical Gradient Descent algorithm. But that doesn’t matter all that much because the path taken by the algorithm does not matter, as long as we reach the minima and with significantly shorter training time.

STOCHASTIC GRADIENT DESCENT CLASSIFIER



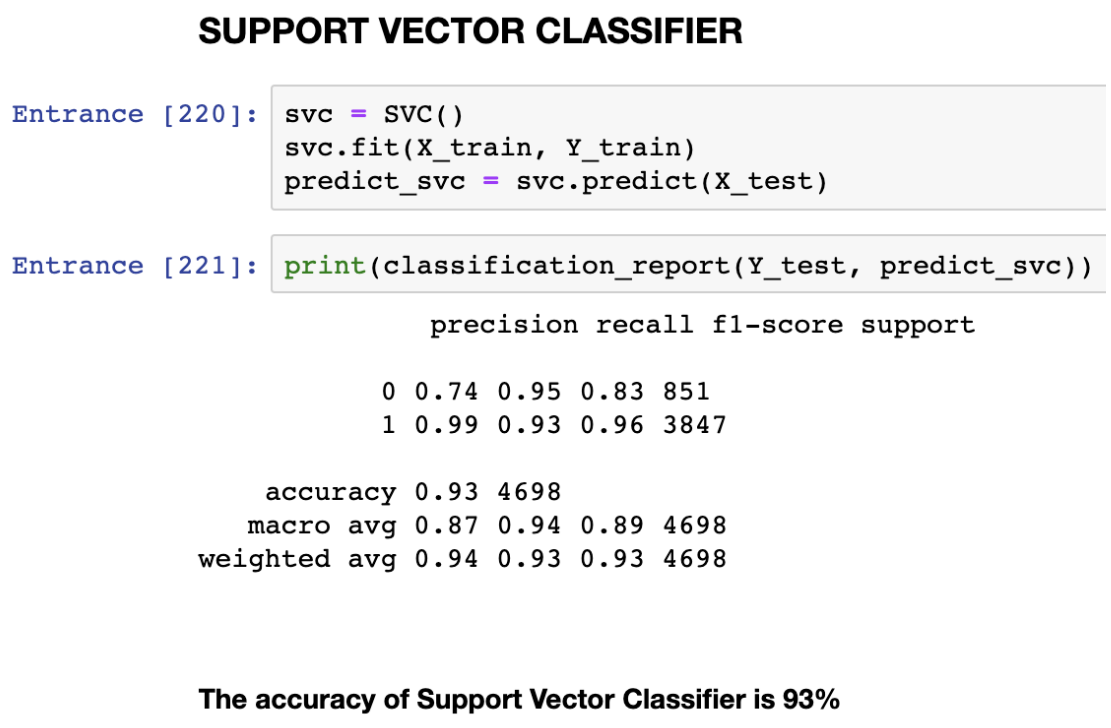
```
Entrance [218]: sgd = SGDClassifier(penalty=None)
                sgd.fit(X_train, Y_train)
                predict_sgd = sgd.predict(X_test)

Entrance [219]: print(classification_report(Y_test, predict_sgd))
                           precision recall f1-score support

                        0 0.74 0.95 0.83 851
                        1 0.99 0.93 0.96 3847

                 accuracy 0.93 4698
                macro avg 0.87 0.94 0.90 4698
             weighted avg 0.94 0.93 0.93 4698
```

The accuracy of Stochastic Gradient Descent Classifier is 93%

3. SUPPORT VECTOR CLASSIFIER (SVC)

SVM offers very high accuracy compared to other classifiers such as logistic regression, and decision trees. It is known for its kernel trick to handle nonlinear input spaces. It is used in a variety of applications such as face detection, intrusion detection, classification of emails, news articles and web pages, classification of genes, and handwriting recognition.

SVM is an exciting algorithm and the concepts are relatively simple. The classifier separates data points using a hyperplane with the largest amount of margin. That's why an SVM classifier is also known as a discriminative classifier. SVM finds an optimal hyperplane which helps in classifying new data points.



SUPPORT VECTOR CLASSIFIER

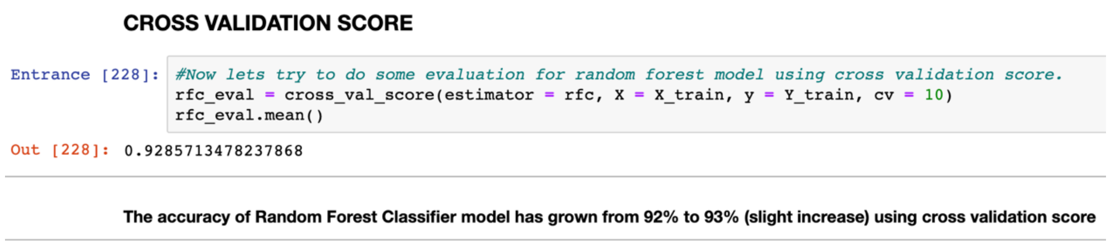
```
Entrance [220]: svc = SVC()
                svc.fit(X_train, Y_train)
                predict_svc = svc.predict(X_test)

Entrance [221]: print(classification_report(Y_test, predict_svc))
```

```
                    precision recall f1-score support

                 0 0.74 0.95 0.83 851
                 1 0.99 0.93 0.96 3847

          accuracy 0.93 4698
         macro avg 0.87 0.94 0.89 4698
      weighted avg 0.94 0.93 0.93 4698
```

The accuracy of Support Vector Classifier is 93%

4. CROSS VALIDATION SCORE

Learning the parameters of a prediction function and testing it on the same data is a methodological mistake: a model that would just repeat the labels of the samples that it has just seen would have a perfect score but would fail to predict anything useful on yet-unseen data. This situation is called overfitting. To avoid it, it is common practice when performing a (supervised) machine learning experiment to hold out part of the available data as a test set **X_test**, **y_test**. Note that the word "experiment" is not intended to denote academic use only, because even in commercial settings machine learning usually starts out experimentally. Here is a flowchart of typical cross validation workflow in model training. The best parameters can be determined by grid search techniques. In scikit-learn, a random split into training and test sets can be quickly computed with the train_test_split helper function.



CROSS VALIDATION SCORE

```
Entrance [228]: #Now lets try to do some evaluation for random forest model using cross validation score.
                rfc_eval = cross_val_score(estimator = rfc, X = X_train, y = Y_train, cv = 10)
                rfc_eval.mean()

Out [228]: 0.9285713478237868
```

The accuracy of Random Forest Classifier model has grown from 92% to 93% (slight increase) using cross validation score

RESULTS

Algorithms	No. Of Iterations	Cost	Speed
Gradient Descent	500	0.24	Slow
	1000	1.00	Fast

Algorithms	Accuracy %
Random Forest Classifier	92%
Stochastic Gradient Descent Classifier	93%
Support Vector Classifier	93%
Cross Validation Score (implemented on Random Forest Classifier model to increase the accuracy %)	93%

CONCLUSION

As a result of performing multiple algorithms on the dataset, it is evident that obtained accuracy (93%) is equivalent to the other models i. e, Random Forest Classifier, Stochastic Gradient Descent Classifier and Support Vector Classifier and the models are by functioning well with the fetched data. In addition, the Cross Validation Score technique has increased the accuracy of the Random Forest Classifier model from **92%** to **93%**.