Case Study: Product returns prediction

Problem Statement:

Customers send back a substantial part of the products that they purchase online. Return shipping is expensive for online platforms and return orders are said to reach 50% for certain industries and products. Nevertheless, free or inexpensive return shipping has become a customer expectation and de-facto standard in the fierce online competition on clothing, but shops have indirect ways to influence customer purchase behaviour. For purchases where return seems likely, a shop could, for example, restrict payment options or display additional marketing communication.

Data Set:

TrainingData_v1 - To be used for training and testing

TestingData_For_Candidate - To be used for predicting the output (this will be compared against the results that we have retained)

1. Introduction:

With the rapid growth of e-commerce, the cost to handle returned online orders also increases significantly and has become a major challenge in the ecommerce industry. Accurate prediction of product returns allows organization to prevent problematic transactions in advance.

This approach consists of two sub problems, which needs to be solved, so that the company can maximize its profit. First how to handle the large-scale data sets involved in this problem, typically consisting of huge numbers of customers and thousands of products so that data will be used efficiently and effectively to predict the product returns.

Second the good machine learning prediction model needs to be used, which can predict whether customer is likely to return specific item or not. Nudging prospect not to buy a certain product, in itself means giving up on potential income and if not predicted correctly this might give bigger loss to company than not having any model in place at all.

The purpose of this case study is to analyze the provided data regarding historical purchase decisions of online shop users, use machine learning techniques on it and based on this predict if customer is likely to return the item in the future or not.

The first part of the case study represents general overview of the data, and data preparation & Feature selection. In second part the prediction models are discussed and the performance is evaluated. Lastly the conclusions are provided.

2. Overview of Data:

As the first step of data analysis is done in order to get overall understanding of data. The data consists of thirteen fields with user decision namely: order_item_id, order_date, delivery_date, item_id, item_size, item_color, brand_id, item_price, user_id, user_title, user_dob, user_state, user_reg_date, user_decision.

After observing the dataset, it can be separated into important and non-important fields that can be later used in the prediction model. There are few data fields such as price, item size, item color, and user title, etc. which may play significant role in decision making.

It is also observed that this dataset consists of three ids such as order id\item id\user id. These Id fields seem to be less informative to the prediction model. This dataset also consists of four date fields from which new important data fields can be generated such as age of user on the date of order, delivery time and for how long user is a registered member.

After checking all the data there are suspicious dates for which delivery date is less than order date.

There are few empty fields in the dataset. There are people who intentionally did not wanted to share their information. Later, these fields can be modified and exclude such values, or replace them with some logical ones.

3. Data preparation & Feature selection:

The insight from data analysis step is considered for feature creation and replacing current values. The new important fields are created which contain relevant information and model could benefit from it are as follows:

Age: age of user at order time is crucial information which is computed from DOB (date of birth) on the date of order.

Delivery time: one of the important fields is the delivery time which can be computed from date of order to date of delivery.

Register user time: how long user is a registered member is also a significant factor which can be found from user registration date to order date.

New fields regarding dates: New variables are generated from date fields, which later can turn out to be useful such as day, month and year.

Categorical data field: It can be assumed that item size and color are important factors for customers; both can play a crucial role in decision making. For both the variables, one hot encoded technique is used.

After creating new features and mapping the existing feature, some of the variables may not be much relevant or less informative. Irrelevant variables or correlated variables could deteriorate the performance of classifier. On the other hand, large number of input variables leads to increase in the computational cost and risk of over-fitting as well as noise. Permutation feature importance technique is used to transform the feature space from higher dimension to lower dimension by removing redundant and highly correlated variables from feature space.

3.1. Feature selection:

There are various feature selection or dimension reduction techniques but many of them do not meet the following requirements such as detect redundant feature, and inter correlation between input variable, and also to find the non-linear relation between the response variable and features. **Permutation feature importance technique (PM)** using Random Forest is used as the feature selection model to find the feature relevance score, due to its robustness.

The concept of Permutation importance technique is really straightforward: the importance of a feature is measured by calculating the increase in the model's prediction error after permuting the feature. A feature is "important" if shuffling its values increases the model error, because in this case the model relied on the feature for the prediction. A feature is "unimportant" if shuffling its values leaves the model error unchanged, because in this case the model ignored the feature for the prediction. This is especially useful for non-linear or opaque estimators. This technique benefits from being model agnostic and can be calculated many times with different permutations of the feature. This method will randomly shuffle each feature and compute the change in the model's performance. The features which impact the performance the most are considered to be the most important one.

4. Prediction models tuning and selection:

The models used for predicting the data are Random forest, Support Vector Machine (SVM) and XGboost, due to their robustness. All the models used here are based on different concepts such as Random forest is an ensemble learning technique while XGBoost is normally used to train gradient-boosted decision trees and other gradient boosted models. On the other side, SVM is a structure risk minimization based classifier.

Before giving cleaned data to models it is first mandatory to convert them to the numeric fields, as some models do not accept factor variables. There are two possibilities to solve this issue. First numerical category encoding technique and second dummy variables are used. Once the data is encoded, it is partitioned.

After partitioning data is given to feature selection model to fit the model and find the relevant score. Once the relevant score is obtained, important feature can be selected based on various metrics such as mean score, sorted features based on the score and chosen fixed number of features; here the features are selected based on the criteria that the relevant score should be greater than zero **(feature score> threshold)**.

After selecting the optimal feature subset, the data is given to SVM, random forest and XGboost. To find best parameters, models are tuned. After tuning is finished, performances of models are evaluated using AUC and accuracy.

5. Results & Observation

To make model complete and predict best AUC possible, the different learning techniques are used and various experiments have been performed on this dataset, all the variables are encoded as described in section 3 (Data preparation & Feature selection) except the following three categorical variables ['item_size', 'item_color', 'user_title']. For these three variables, two encoded techniques are used: One hot encoded (get_dummies in pandas) and numerical category encoding technique (astype('category').cat.codes) are used.

Results with One hot encoded technique: Dummy variables are created for item size, item color, and user title and created an input vector of size 203. This input vector is passed to permutation feature importance technique (PM) to compute the relevance score. As depicted in Fig.1, the features which got the higher score are used to predict the user decision. The features are selected based on the criteria that the relevant score should be greater than zero **(feature score> 0)**.

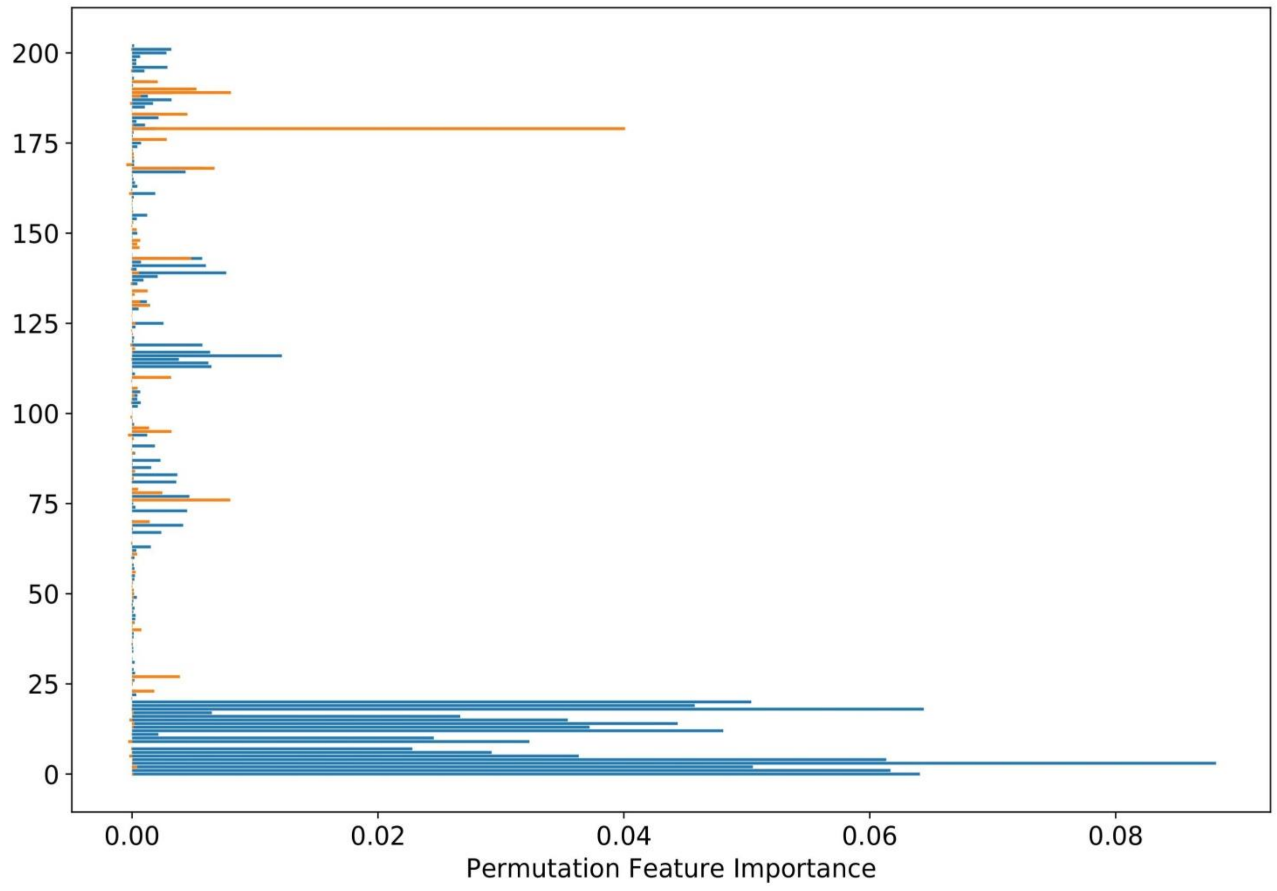

Fig.1

To train and fine tune the parameters, the dataset is partitioned into 60:40 ratio, where 60 percent data samples are used to train and fine tune the parameters while 40 percent samples are used to evaluate the performance of prediction models.

Table 1

AUC (%)	Random forest	XGboost	SVM
With PM	68.54	68.35	63.56
Without PM	66.45	67.85	62.74

As shown in Table 1, Random forest and XGBoost gives better performance over Linear SVM; and PM feature selection improved the performance of all prediction models.

Numerical category encoding technique: Numerical categories are created for item size, item color, and user title and created an input vector of size 24. This input vector is passed to permutation feature importance technique (PM) to compute the relevance score. The features are selected based on the criteria i.e., **feature score> 0**. As depicted in Fig.2, 21 features are used to predict the user decision.

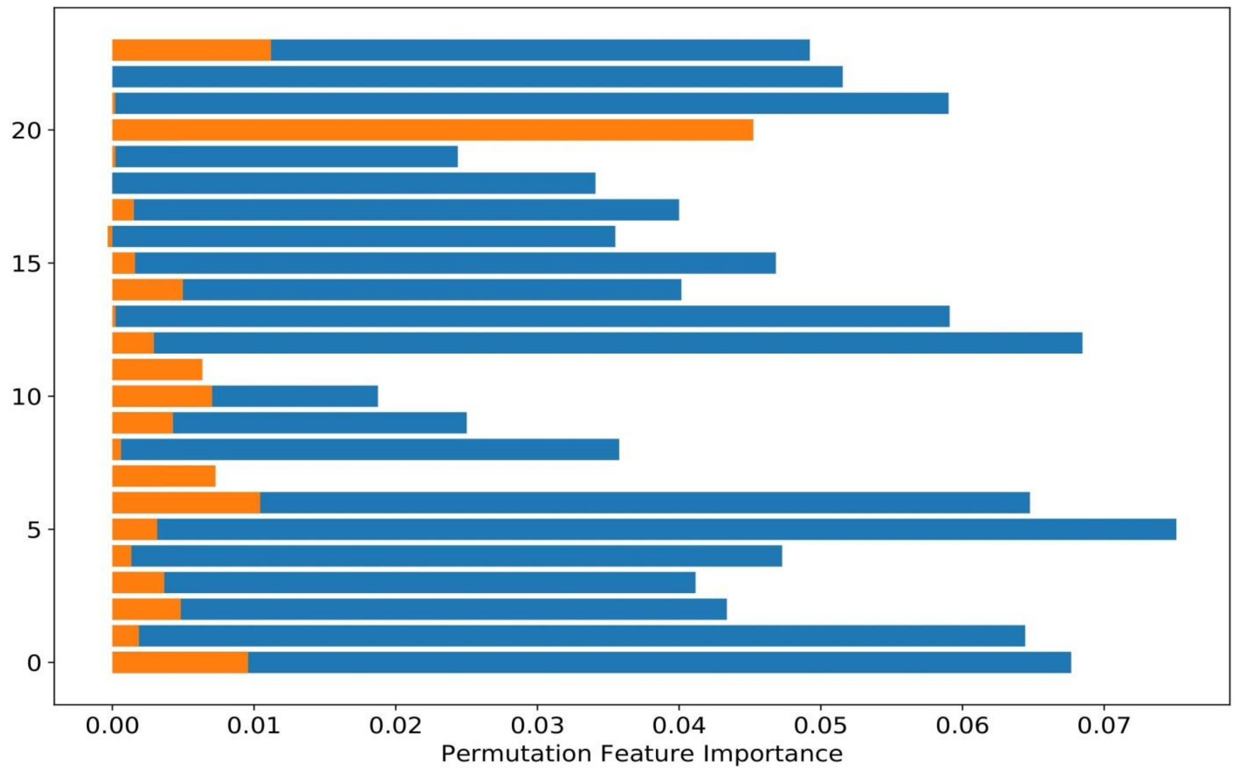

Fig. 2

The prediction models are trained and fine-tuned using the same setup used for one hot encoded technique.

As shown in Table 2, similar to one hot encoded technique, Random forest and XGBoost gives better performance over Linear SVM and PM feature selection improved the performance and interpretability of all prediction models.

Table 2

AUC (%)	Random forest	XGBoost	SVM
With PM	**73.18**	**74.29**	**67.03**
Without PM	66.54	68.85	63.42

The detailed results of prediction models with PM feature selection Technique

- **XGBoost Prediction model:**

Roc		**0.742979**	**Threshold**		**0.420612**
	Accuracy	True positive	False Positive	False Negative	True Negative
Best threshold	**0.668897**	**10152**	**7112**	**3476**	**11238**
thresh_0.5	0.671774	11795	5469	5027	9687
thresh_0.4	0.665176	9703	7561	3146	11568
thresh_0.3	0.647289	7676	9588	1691	13023
thresh_0.25	0.633185	6665	10599	1131	13583

- **Random Forest:**

Roc		**0.731835**	Threshold		0.455836
	Accuracy	True positive	False Positive	False Negative	True Negative
Best threshold	**0.657733**	**10217**	**7047**	**3898**	**10816**
thresh_0.5	0.659922	11815	5449	5426	9288
thresh_0.4	0.644787	8408	8856	2503	12211
thresh_0.3	0.614329	5846	11418	915	13799
thresh_0.25	0.596222	4780	12484	428	14286

- **Linear SVM Prediction model:**

Roc		**0.67036373**	Threshold		**0.444196255**
	Accuracy	True positive	False Positive	False Negative	True Negative
Best threshold	**0.612046**	**7657**	**9607**	**2799**	**11915**
thresh_0.5	0.608887	11850	5414	7093	7621
thresh_0.4	0.584214	4874	12390	906	13808
thresh_0.3	0.555632	3083	14181	29	14685
thresh_0.25	0.555069	3036	14228	0	14714

There are few challenges in this case study, in the preprocessing phase, one of the main problems which occurred is inconsistency between dates, and e.g., delivery date is less than order date. Real situation into consideration could have been a great help, as people most of the times have direct affection to the specific brand and specific item. Despite this fact current models provide the accuracy of 74% which itself is great result, considering that decisions needed to be made specifically on features of items (such as color, price, size).

Note:

1. Please refer to the Readme file for code details.

2. The user decision prediction on test data is saved in “results: directory (xgb_predict_results.csv, rf_predict_results.csv, svm_predict_results.csv).

3. SVM and XGboost artifacts are saved in “data” directory. Random forest artifact is not saved due to its size.

References:

[1] C. Cortes, V. Vapnik, Support-vector networks, Mach. Learn. 20 (3) (1995) 273–297.

[2] L. Breiman, Random forests, Mach. Learn. 45 (1) (2001) 5–32.

[3] R. Genuer, J.-M. Poggi, C. Tuleau-Malot, Variable selection using random forests, Pattern Recogn. Lett. 31 (14) (2010) 2225–2236.

[4] Fisher, Aaron, Cynthia Rudin, and Francesca Dominici. “Model Class Reliance: Variable importance measures for any machine learning model class, from the ‘Rashomon’ perspective.” http://arxiv.org/abs/1801.01489 (2018).

[5] XGBoost: A Scalable Tree Boosting System. arXiv:1603.02754