FASHION IMAGE CLASSIFICATION: USING MACHINE LEARNING

Tajudeen Abdulazeez
IST718, Big Data Analytics
School of Information, Syracuse University
toabdula@syr.edu

INTRODUCTION

According to the publication by Tara Johnson(https://www.cpcstrategy.com/blog/2019/03/future-of-fashion/), Annual AI spending is predicted to grow to $7.3 billion by 2022. Machine learning is in the uptrend in retails industries, so it's no surprise that AI is becoming integral part of the fashion industry. There are different ways in which AI is transforming the future of fashion and the five major ways is listed below:

1. Manage Inventory: Machine learning algorithms use historical data to make predictions and choices. AI-powered tools for demand forecasting use these algorithms to help solve the age-old industry pain point. These tools can help retailers reduce forecasting errors by up to 50% while reducing inventory by 20-50%.
2. Connect with customers: Many fashion retailers use AI chatbots, also called smart assistants, to connect with customers and provide product recommendations. This scalable method of customer service can help retailers save money on customer service staff while building customer loyalty.
3. Tailor Recommendation: In order to keep costs low, brands need to better predict customer preferences by gathering and analyzing purchase data. Using this data alongside AI and machine learning allows fashion retailers to create clothes that customers want to buy.
4. Reduce Returns: AI can help retailers personalize the shopping experience, leading customers to make more informed purchase decisions. In addition to improving customer satisfaction, this reduces the return rate and saves retailers money.
5. Improve Product Discovery: Another AI-powered retail trend, visual search makes it easier than ever for shoppers to discover and purchase the products they want. Shoppers simply snap and upload a photo of the product they want, then AI identifies the pictured product (or similar ones) across multiple sites and retailers.

The aims of this project is to evaluate different machine learning approach using the accuracy and compute time in training and prediction to select a machine learning with the best performance based on the mentioned criteria and the selected machine learning hyperparameters is optimized for possible improvement in model accuracy in classifying fashion items.

ANALYSIS AND MODEL

This analysis is carried out using python programming language with the following packages:

NumPy: For numeric computing

Pandas: For tabular data manipulation

Matplotlib: For visualization

Struct: For unpacking the data

Seaborn: For data visualization

Sklearn: For modeling

Keras: for modeling.

About the Data

The dataset used for this project is from Zalando research(https://github.com/zalandoresearch/fashion-mnist). The datasets contain 10 labels encode using numeric from 0 -9 to represent each of the labels.

Label	Description
0	T-shirt/top
1	Trouser
2	Pullover
3	Dress
4	Coat
5	Sandal
6	Shirt
7	Sneaker
8	Bag
9	Ankle boot

Table 1: The description of each label in the fashion datasets

Below is the visualization of some randomly selected data as shown in the figure below

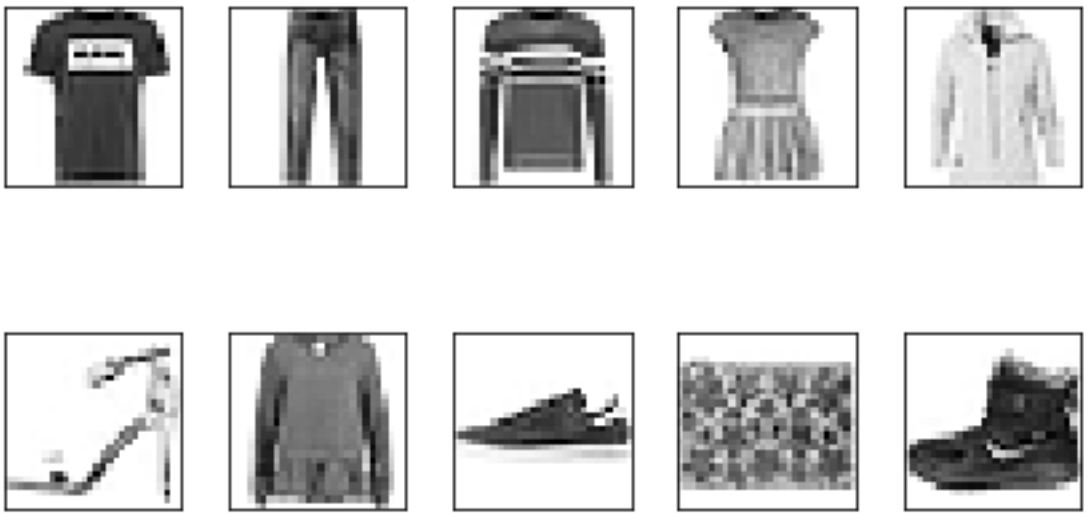

Figure 1: Some randomly selected fashion items from the datasets.

The training data has 60,000 rows with a column dimension of 784 and the test set contain 10,000 rows with column dimension of 784. The training data is used to train the model, taken notes of the training accuracy and the time it takes to train the model. The test set which the algorithm has never seen is

used to test the performance of the model using accuracy as the evaluation metrics. The prediction time is also recorded in other to compare the compute time and accuracy of each model used in this analysis.

Below is the table showing the accuracy of the training and test set for reach model and the compute time.

	Model	Prediction Time	Test Accuracy	Training Accuracy	Training Time
0	GaussianNB	1.774898	0.5856	0.587783	1.705738
1	RandomForest	0.093734	0.8546	0.995150	12.764866
2	GradientBoostingClassifier	0.656541	0.8681	0.903650	4212.935617
3	nnMLP	0.109553	0.8688	0.902550	282.424564

Table 1: Accuracy results for training and test set for each model and the compute time.

The Gaussian Naïve Bayes has lower training and test accuracy of around 58% with training time of 1.7sec and predicting time of around 1.7sec. Random forest has a training accuracy of 99% and a test accuracy of of 85%, there might be an element of overfitting of the training set. The training takes about 12.7sec and take about 0.09sec to make prediction. The prediction time is faster compared the Gaussian Naïve Bayes, but random forest takes longer training time. Gradient Boosting takes longer time to train with improved test accuracy of 86.8%. The deep learning with 50 hidden layer takes about 282.4sec, the highest training time with test accuracy of about 86.8%.

Random Forest

Random forest is an ensemble learning method for classification, regression and other tasks that operates by constructing a multitude of decision trees at training time and outputting the class that is the mode of the classes (classification) or mean prediction (regression) of the individual trees. Random decision forests correct for decision trees' habit of overfitting to their training set

Benefits of tree-based model

- Works for both classification and regression
- Handles categorical features naturally
- No assumption of distributions
- Can handle non-linear interactions
- No need for feature scaling / transformation
- Handles missing values in the variables

Advantage of Random Forest

- Built in cross validation (OOB Scores)
- Built in Feature Selection (implicit)
- Feature importance
- Default hyper parameters are great and Works well "off the shelf"
- Minimum hyper parameter tuning
- RF natively detects interactions
- It's parametric (you don't have to make any assumptions of your data)

Disadvantage of Random Forest

- RF is a black box (It's literally a function of 1000 decision trees)
- It doesn't tell you "how" the features are important

Parameters of Random Forest that can improve its performance

- max_depth: The depth size of a tree
- n_estimators: The number of trees in the forest. Generally, the more trees the better accuracy, but slower computation.
- max_features: The max number of features that the algorithm can assign to an individual tree. Try ['auto', 'None', 'sqrt', 'log2', 0.9 and 0.2]
- min_samples_leaf: The minimum number of samples in newly created leaves. Try [1,2,3]. If 3 is best, try higher numbers.

Parameters that can improve the speed of Random Forest

- n_jobs: Determines the amount of multiple processors should be used to train/test the model. Always use -1 to use max cores and it'll run much faster
- random_state: Set this to a number (42) for reproducibility. It's used to replicate your results and for others as well.
- oob_score Random Forest's custom validation method: out-of-bag prediction

Model Optimization

Turning the hyperparameters of random forest for possible improvement of model performance.

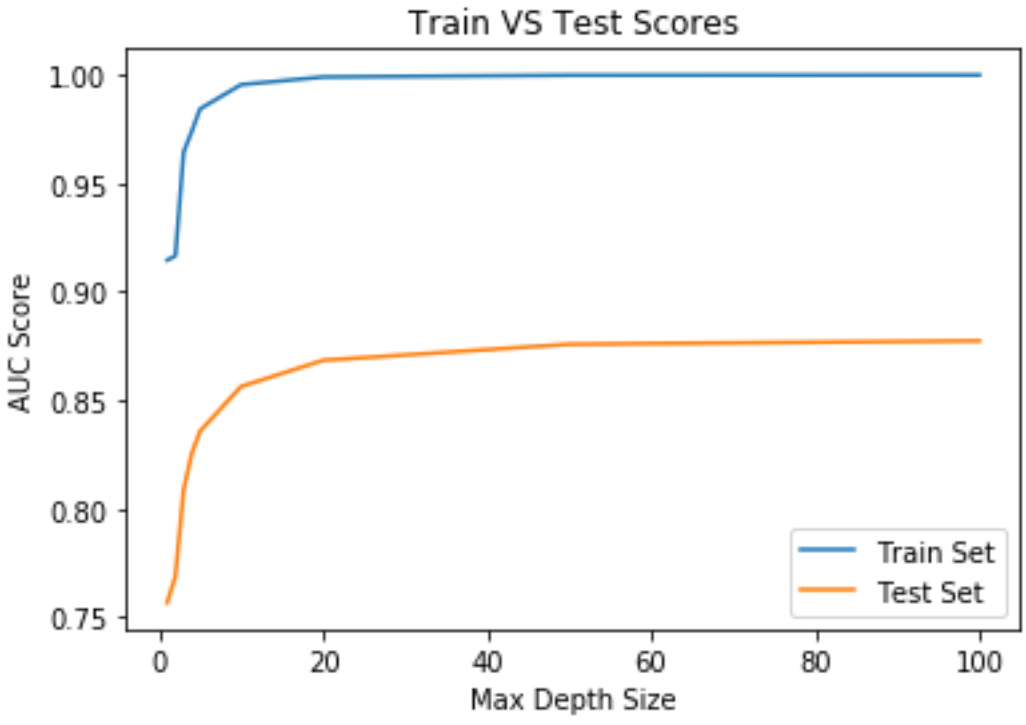

Figure 2: Max depth vs Accuracy of the model on training and test set. (Max depth = 50)

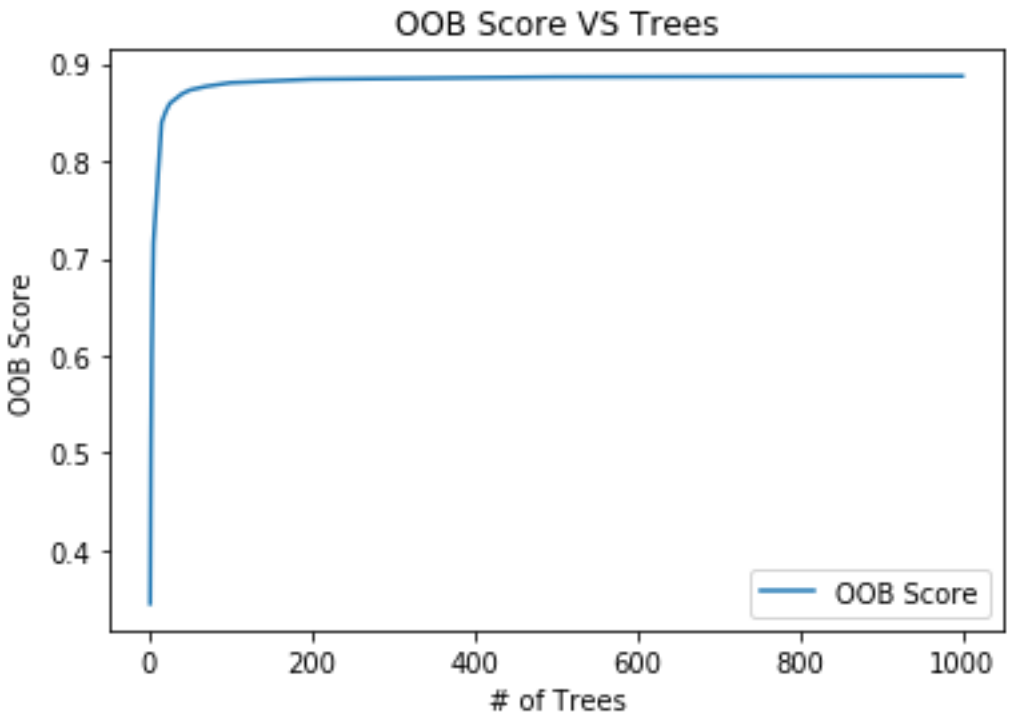

Figure 3: OOB score with number of trees (n_estimators =100)

Generally the more trees the better. You'll generalize better with more trees and reduce the variance more. The only downside is computation time.

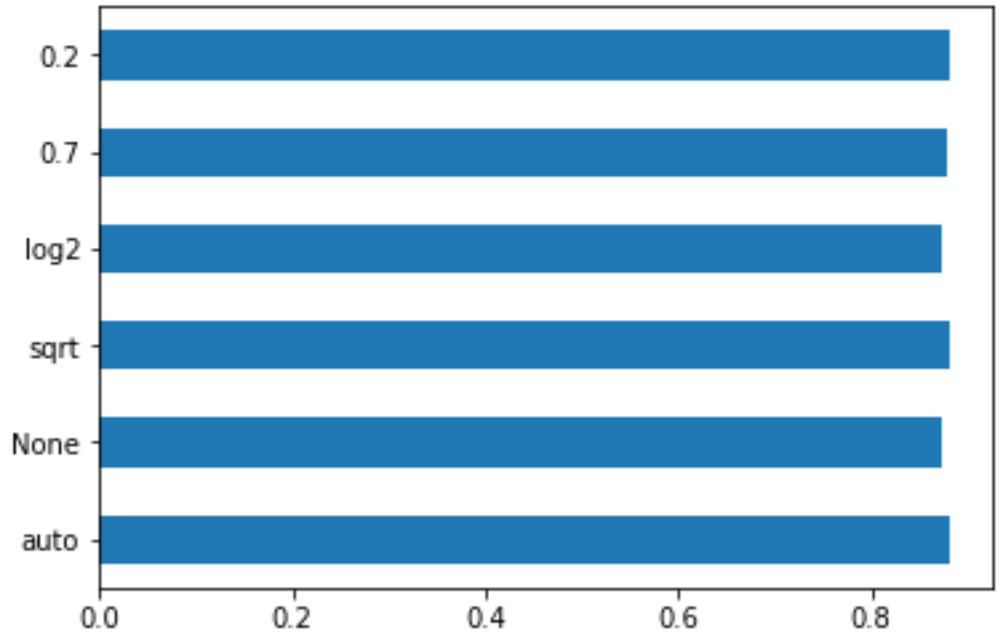

Figure 4: Max features of auto has the best performance with accuracy of 88.05%

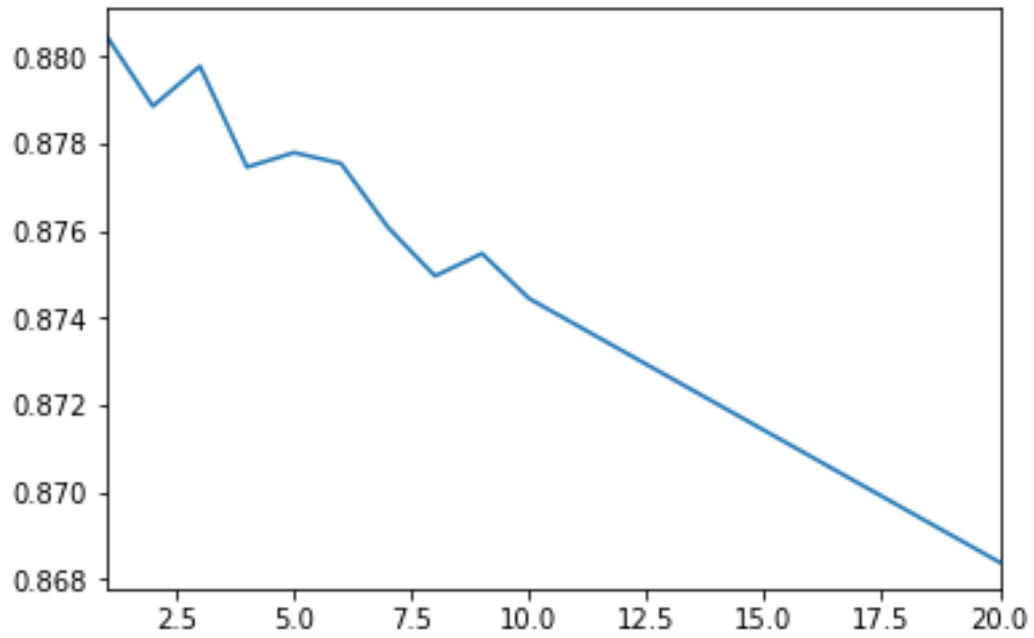

Figure 5:The accuracy is on the y-axis and min_leaf is on the x-axis. The accuracy drop as the min_leaf increases. The optimal min_leaf selected for the final model is 1.

Final model parameters

RandomForestClassifier(n_estimators=100, oob_score=True, n_jobs=-1, random_state=42, max_features ="auto", min_samples_leaf=1 , max_depth=50)

Training Time	84 sec
Training Accuracy	87.6%
Prediction Time on Test set	0.5 sec
Test Accuracy	87.6 %

Conclusion

Based on the results of the analysis, Random forest is chosen for classifying fashion image based on the following:

1. Random forest can handle nonlinear interaction and no assumption of distribution is needed.
2. Random Forest did not need feature scaling/transformation and can handle missing values in the variables.
3. Test accuracy of 87.6% on the test set and it is also faster to train compared to Neural network.
4. Random forest natively detects interaction and required minimal hyperparameters tuning.

Source Code

https://github.com/toraaglobal/fashion-mnist