## Algorithm Selection

The three classification algorithms I have chosen to use are the Random Forest Classifier, Logistic Regression and the K-Nearest Neighbour (KNN) algorithm. Other options e.g. SVM were decided against due to very high training times and Naive Bayes due to its lack of robustness.

### Random Forest:

A Random forest classifier was used for a few key reasons. Random Forest often produces strong results for classification, more so than a decision tree classifier or even a bagged set of decision trees. This is because although Random Forest is essentially a bagged decision tree model the Random Forest classifier is split on a random subset of features which decorrolates the trees within the model. The hyper-parameters which were focussed on were the number of estimators, the max features and the max depth. The number of estimators refers to the number of trees in the forest, for data which is not noisy overfitting is very unlikely for the random forest classifier, therefore if the number of estimators is larger than necessary the cost is longer and more computationally demanding training rather than a reduction in model performance. The max features hyper-parameter refers to the number of features to consider when the model looks for the best split. The max depth hyper-parameter refers to how deep the tree's are in the model; the deeper the tree the more splits it will have and the more information it will be able to perceive from the data.

### Logistic Regression:

Logistic regression was chosen as it is a very simple algorithm to implement with few hyperparameters to tune. Although binary classification is the preferred use case, logistic regression still performs quite well on multi-class classification problems. The hyper-parameters which will be focussed on are the maximum number of iterations and the C hyperparameter which denotes the inverse of regularization strength. C is a penalty term which can help regulate against overfitting. The max number of iterations refers to the number of iterations over the data the solver will allow for it to converge.

### KNN:

KNN was chosen as it has some inherent advantages to classifiers such as logistic regression, one of these advantages is that it requires no assumptions about the data. The data can be of any kind as opposed to logistic regression where the data needs to be linearly separable. The hyper-parameters which were focussed on were the number of neighbours and the algorithm used to choose the neighbours. The number of neighbours refers to the number of 'voters' will be used to classify a certain image, if the number of neighbours is increased the decision boundary will be smoother with lower variance but potentially higher bias. A higher number of neighbours will also require more computation as more distances need to be calculated. The algorithm used determines how the neighbours are chosen. The optimum algorithm will depend on the data size and computation available as some algorithms have much higher time complexity than others.

## Methodology

In order to optimise the hyperparameters over the three models a Grid search function was used. This enables the model to test all of the potential combinations of a dictionary of different hyperparameters in order to determine which combinations produced the highest model performance. However, due to computational limitations, not every possible value is able to be tested against each other as this is just not feasible. The grid search therefore used a set of values for each hyperparameter which was already filtered to remove unnecessary values e.g. even values for the number of neighbours for KKN to ensure majority voting. The values which would also cause too high a training time were also not used e.g. a cap of 1500 for the number of estimators in the random forest classifier. The performance measure which was used was the test accuracy of the model. This alongside a sci-kit learn classification report for each model to observe which classes were consistently correctly identified and which ones may have had persistent problems.

**Random Forest:**

The grid search values for the random forest classifier were as follows: number of estimators: (10, 100, 1000, 1200, 1500) of which 1500 produced the highest test accuracy when paired with the other optimum hyperparameters. The gridsearch values for the max number of features were: (17, 21, 25) of which 25 reported the highest test accuracy. The gridsearch values for the max number of features were: (15, 20, 25) of which 25 reported the highest test accuracy. These results suggest that the model, if more computational power was available, may benefit from using larger values for potentially all of the hyperparameters.

**Logistic Regression:**

The gridsearch values for the value of C were: (0.1, 1, 10, 100) of which 1 reported the highest test accuracy. The gridsearch values for the number of iterations were: (100, 1000, 10000) of which 100 reported the highest test accuracy. The logistic regression model requires the least tuning out of the three classifiers which were used and the values which produced the highest test accuracy were the default options in the sci-kit learn implementation of logistic regression which was used. However, as there is no 'one size fits all' when optimising parameters for different datasets this step was still necessary.

**KNN:**

The gridsearch values for the number of neighbours were: (1, 3, 5, 7, 9, 11) of which 5 reported the highest test accuracy. The gridsearch options for the algorithm implemented were ball tree and kd tree of which ball tree reported the highest test accuracy.

## Results

**Random Forest:**

The random forest classifier produced the best test accuracy at its optimum hyperparameter settings out of the three classifiers chosen. The accuracy score was 0.8838. This was significantly stronger than both of the other classifiers. Although almost all of the classes performed very well there were a couple of classes which had notably lower precision scores than others. The 6th class, which corresponds to Shirts, had the lowest precision score at 0.75 along with classes 0, 2 and 4 which had precision scores in the low 80's. This makes sense when it is considered what these classes correspond to: T-shirt, Pullover and Coats respectively. All of these items from a 2D perspective look similar, are mostly the same shape and are understandably much more alike than a shirt would be to a shoe.

**Logistic Regression:**

The logistic regression classifier produced a test accuracy of 0.8509, the lowest of the three classifiers. This was expected as it is a relatively simple model in comparison to the others with lower time complexity. In a similar, but more pronounced, fashion the logistic regression classifier struggled slightly to classify the similar classes of 0, 2, 4 and 6 (T-shirts, Pullovers, Coats and Shirts).

**KNN:**

The KNN classifier produced a test accuracy of 0.8540, slightly higher than logistic regression but not significantly. An interesting result was observed in the classification report of three KNN classifiers; the KNN classifier struggled the most with classifying the problematic classes: 0, 2, 4, 6. with precision scores of 0.77, 0.75, 0.79, 0.68 respectively. However in many of the other classes the KNN algorithm performed better than the random forest classifier, for example in class 5 (Sandals) it achieved a precision score of 1.

## Conclusion

In conclusion, the best performing random forest classifier was superior to the other chosen classifiers by a large margin. A reason why this might be is its ability to handle high-dimensional spaces and large numbers of training examples. In addition to this performance benefit the training time was significantly faster than the KNN classifier.