COMPSCI 5014 - Machine Learning (Level M)

Assessed Coursework – UoG Kaggle Competition 2018

Submission Report

School of Computing Science

Presented by

Ernst Werner

MSc Data Analytics

2383746W

Presented to

Dr Ke Yuan

November 26th, 2018

Table of Contents

List of Figures

List of Tables

1. Literature Review

Scholarship within the AI and ML literature specifically focusing on natural language processing models (NLP) and word/sentence embedding has recently paid attention to contributions by the team of Google researchers led by Tomas Mikolov (including Kai Chen, Greg Corrado, Jeffrey Dean and Ilya Sutskever) for their original 2013 development of the '*Word2vec'* model, used among other things, to reconstruct linguistic contexts of words, i.e. to produce word embeddings. More broadly, these researchers are widely recognized for pushing the research frontier by combining novel methods to complement and expand pre-existing types of neural network language modelling (NNLM) techniques while optimizing (lowering) computational cost and improving on the accuracy and efficiency of results when using large data sets (corpora).

In a first, extensively circulated paper on the "*Efficient Estimation of Word Representations in Vector Space"* (Mikolov, T. et al 2013) introduce two new log-linear model architectures—namely a '*Continuous Bag-of-Words (CBOW)*' model and a '*Continuous Skip-gram Model'* — that can be used for learning distributed representations of words whilst minimizing computational complexity and being trained on larger amounts of data much more efficiently than standard neural network models permitted. The proposed models are trained using stochastic gradient descent and backpropagation. Whereas the CBOW architecture they present predicts a current word based on its context, their Continuous Skip-gram model predicts surrounding words given the current word, that is, it maximizes the classification of a word based on another word in the same sentence.

Their work responded to the acknowledgement that previously proposed architectures had only been successfully trained on more than a few hundred of millions of words (with modest word vectors' dimensionality in the 50 - 100 range). Realizing that the simple scaling up of basic techniques had not yet resulted in significant progress to process larger corpora, the techniques they introduce are geared to be used for learning high-quality word vectors from huge data sets with billions of words (in the paper ~ 1.6 billion words dataset). Mikolov, T. et al 2013 focus on the first step of NNLM techniques where the word vectors are first learned using the neural network with a single hidden layer and they try to maximize the accuracy of vector operations by ensuring that their proposed new model architectures preserve the linear regularities among words. While the well-known well-known Latent Dirichlet Allocation (LDA) and Latent Semantic Analysis (LSA) have been frequently used

to estimating continuous representations of words they focus on expanding models of distributed representations of words learned by neural networks because the latter perform significantly better than LSA for preserving linear regularities among words and are much less computationally expensive when using large corpora than LDA.

Moreover, responding to a prior research limitation which recognized that simple standard NLP techniques such as N-grams neglect the notion of words' similarity by treating them as atomistic units represented by vocabulary indices, Mikolov, T. et al 2013 also design a comprehensive test set for measuring both syntactic and semantic regularities to measure the quality of word vectors. Their test defined five types of semantic questions (comprising about 8869 questions) and nine types of syntactic questions (comprising about 10675 questions). They first created a manual list of similar word pairs, then a large list of questions was formed by connecting two-word pairs, and lastly, they evaluated the overall accuracy for all question types, and for each question type separately (semantic, syntactic). Their results echoed their prior observations on the quality of vector representations which posited that similar words not only tend to appear close to each other but that words can have multiple degrees of similarity (see T. Mikolov, W.T. Yih, and G. Zweig 2013) and in particular they showed that similarity of word representations goes beyond simple syntactic regularities.

In "*Distributed Representations of Words and Phrases and their Compositionality"*—a subsequent, closely related paper — Mikolov T. et al (2013) develop extensions to their previously introduced continuous Skip-gram model in order to improve on the training speed and on the quality of the vectors deriving from it. Their continuous Skip-gram model architecture predicts words within a certain range before and after a given current word by using each current word as input to a log-linear classifier with continuous projection layer. As briefly noted above, specialists in the field of neural network based language models (RNNLM) have recognized as a forefront efficient method for learning high-quality distributed vector representations that capture a large number of precise syntactic and semantic word relationships because among other things it constitutes a neural network architecture for learning word vectors that do not involve dense matrix multiplications, therefore making it an efficient method of training.

The problematics Mikolov, T. et al (2013) try to address in "*Distributed Representations of Words and Phrases and their Compositionality"* pertains to word representation models' indifference to word order and their inability to represent idiomatic phrases that are not compositions of the individual words (ex: "Boston Globe" a newspaper is not understood as a natural combination of the meanings of "Boston" and "Globe). To respond to these two inherent limitations of word representations Mikolov, T. et al (2013) present several extensions to their original Skip-gram model which result in improved training speed, quality of the vectors, and expressive representation of sentences' meaning.

One key extension they propose allows going from word-based to phrase-based models. Their method to extend from word-based to phrase-based models entailed firstly to identify a large number of phrases through a data-driven approach, and then treating the phrases as individual tokens during the training. Finally, to evaluate the quality of the phrase vectors, they developed a test set of analogical reasoning tasks that contained both words and phrases. Another extension involved the subsampling of frequent words during training in order to counter the imbalance between the rare and frequent words and hence enhance the informational value of training since frequent words (such as "the") usually provide less information value than rare words. Their sub-sampling method accelerated learning (resulted in a significant speedup of around 2x - 10x) and significantly improved the accuracy of the learned vectors of the rare words and of the representations of less frequent words. More precisely, instead of the complex hierarchical *softmax* used in the literature previously, they present a simplified variant of Noise Contrastive Estimation (NCE) for training the Skip-gram model that resulted in faster training and better vector representations for frequent words than the original model.

To perform and test the extensions they rely on an internal Google dataset consisting of various news articles in which they discarded from the vocabulary all words that occurred less than 5 times in the training data, which resulted in a vocabulary of size 692K. Part of their empirical tests and results showed that meaningful compositional results and a non-obvious degree of language understanding can be obtained by using basic mathematical operations (such as simple vector addition) on the word vector representations. Through their approaches for learning representations of phrases, they found that simple vector addition helps to combine word vectors in somewhat meaningful ways. They also showed that by combining such approach with the representation of phrases as single tokens longer pieces of text could be represented in with minimal computational complexity.

As recognized by Mikolov, T. et al 2013, a widely used neural network approach for distributed representation of words is found through Latent Dirichlet Allocation (LDA) models. Blei, D., Ng, A. j., Jordan, M. I. (2003) first introduced the approach as a graphical model for topic discovery; hence, in the realm of natural language processing applications, LDA models are understood as topic models based on generative probabilistic models applied to collections of discrete data such as text corpora.

Before the spread of LDA use, the basic methodology applied by Information Retrieval (IR) researchers onto text corpora was Term Frequency-Inverse Document Frequency (dubbed *tf-idf schema*). While *tf-idf* has been widely deployed in modern Internet search engines, it is not without shortcomings. *tf-idf* reduces each document in the corpus to a vector of real numbers each of which represents in turn ratios of counts. Hence, this technique effectively reduces documents of arbitrary length to fixed-length lists of numbers. Moreover, the *of-IDF* approach also provides a relatively small reduction in description length while revealing little in the way of inter- or intra-document statistical structure. To address the shortcomings of *tf-IDF* IR researchers have proposed alternative dimensionality reduction techniques, including most notably Latent Semantic Indexing (LSI). The latter uses a singular value decomposition of the X matrix to identify a linear subspace in the space of *tf-idf* features that capture most of the variance in the collection.

LSI has been praised for achieving significant compression in large collections. Moreover, Deerwester et al. argue that the features of LSI consisting of linear combinations of the original *of-IDF* features, serve to capture some aspects of basic linguistic notions including synonymy and polysemy. However, given a generative model of text, the benefit of using LSI to recover aspects of the generative model from data has been called into question as once could proceed more directly by fitting the model to data using maximum likelihood or Bayesian methods (Papadimitriou et al., 1998). As an alternative to LSI, Hofmann (1999) developed a probabilistic LSI (LSI) model, known as the aspect model. The LSI approach models each word is generated from a single topic and different words in a document may be generated from different topics. Each document is thus represented as a list of mixing proportions for these mixture components and thereby reduced to a probability distribution on a fixed set of topics, known as the document's "reduced description".

LDA is considered to be similar to probabilistic latent semantic analysis (pLSA) in that documents are understood to be comprised by a set of topics assigned to the given document

via an LDA process. One key difference between pLSA and LDA is that in the latter topic distribution is assumed to be based on sparse Dirichlet priors which encode the intuition that anyone document can only cover a small set of topics which in turn use only a small set of words frequently. Based on this, LDA is thought to allow for better disambiguation of words and a more precise assignment of documents to topics than LSA. In LDA-based models a topic is neither semantically nor epistemologically strongly defined, rather it is identified on the basis of automatic detection of the likelihood of term co-occurrence. While a lexical word may occur in several topics with a different probability of occurrence, it is also recognized that it does so also with a different typical set of neighbouring words in each topic. Because each document is assumed to be characterized by a particular somewhat narrow set of topics LDA has been also considered as similar to the standard bag of words model. Blei, D., Ng, A. j., Jordan, M. I. (2003) results showed that the topic-based representation provided by LDA can be used as a fast filtering algorithm for feature selection in text classification with little reduction in classification performance. In contrast to properties found in Latent Semantic Indexing (LSI), as a probabilistic module, LDA can be readily embedded in complex models. Additionally, LDA can be readily extended to continuous data or other non-multinomial data.

The same type of generative probabilistic topic model underlying LDA and Blei, D., Ng, A. j., Jordan, M. I. (2003) topic model was independently proposed to study population genetics in 2000 by J. K. Pritchard, M. Stephens and P. Donnelly. Similarly, as shown by Buettner, F. et al (2015), similar methods have been applied in Bioinformatics, Imaging and Biotechnology fields. In "*Computational analysis of cell-to-cell heterogeneity in single-cell RNA-sequencing data reveals hidden subpopulations of cells*" Buettner, F. et al (2015) make use of a Single-cell Latent Variable model (solve) to attempt to detect subpopulations of cells for which the existing imaging techniques for cell-subpopulations identification did not provide robust results. The author took scLVM into the account by reanalysing the data processed by single-cell RNA-seq; scLVM provided physiological consequential subpopulations of it which no other algorithm could provide.

Finally, an insightful contribution regarding auto-encoding Variational Bayes methods is found in Kingma, D. P., Welling, M. (2013) where the authors optimized the variational lower bound on the maximum log-likelihood function or posterior (MAP) inference using the stochastic gradient method. The authors then proposed an SGVB (Stochastic Gradient

Variational Bayes) estimator and an AEVB algorithm of the above lower-bound using a Monte-Carlo estimation which could work for all models with any given set of continuous variables. They also employed a re-parametrization trick to get the approximate inference of any random variable by expressing it in their deterministic function (i.e joint Pdf.).

Finally, the authors compared the AEVB algorithm with the wake-sleep algorithm to evaluate two different datasets, MNIST and Frey Face, in terms of the estimated likelihood and the variational lower bound. Their results were considerably faster and produced a better solution in all experiments. It also provided evidence for their regularization of lower bounds. They also compared the AEVB algorithm with Monte Carlo EM for MNIST dataset and showed that AEVB (online) works better.

2. Introduction

As part of the coursework, the goal is to train two given datasets, namely X_train and X_test in order to maximise the overall predictive accuracy of the training data y_train. In satisfaction of the task requirements, two model predictions are provided each for the Regression and Classification problems respectively. Overall, a stepwise approach is followed based on repeated accuracy measurements of different classifiers and regressors as supported by sklearn in Python. Through consecutive parameter adjustments, it was possible to reach higher levels of accuracy in the involved metrics, but also to express more general models. Apart from the analysis, Kaggle submission scores gave a clear indication of highly relevant predictions with satisfactory error margins in the chosen implementation.

3. Exploratory Data Analysis

Before any kind of analysis can be initiated it is of interested to explore the relationship of each features x to the target y. In the regression dataset, PRP is chosen to be the target and the averaged numerical values for the feautured Main Memory and Number of I/O channels is used in order to receive a more concise information. One can clearly see from the scatterplot that most data points are clustered in the lower left bottom of the graphs, indicating that normal linear model assumptions might not hold. It is further visible that none of the features is linearly related to the target, while Machine Cycle Time is clearly inversely related to PRP. Thus, requiring a residual analysis for further exploration.

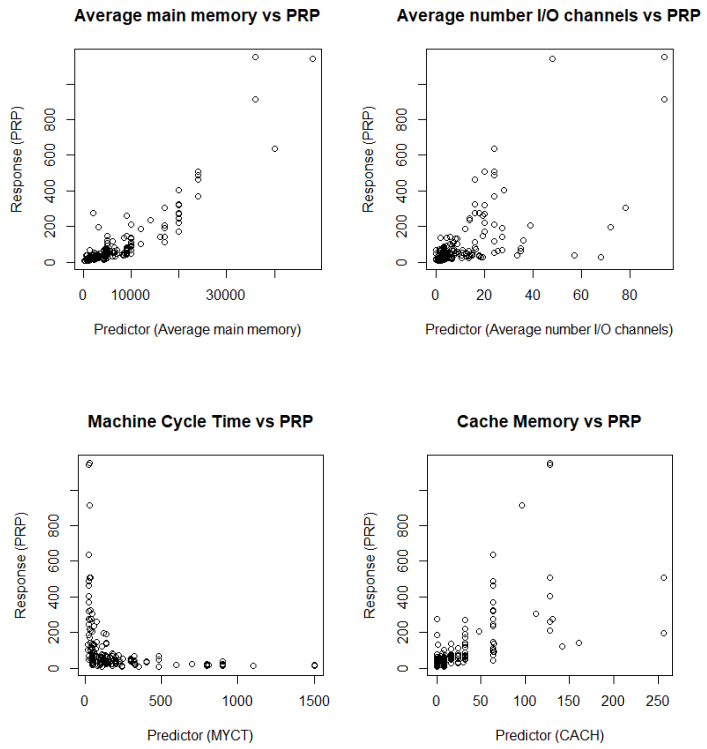

Figure 1: Scatterplot for individual features against the target (PRP)

With a further residual analysis, one wishes to uncover whether the normal linear model assumptions are violated in the given dataset using a simple linear regression for each variable against the target. For graphical display, R studio was used, not Python.

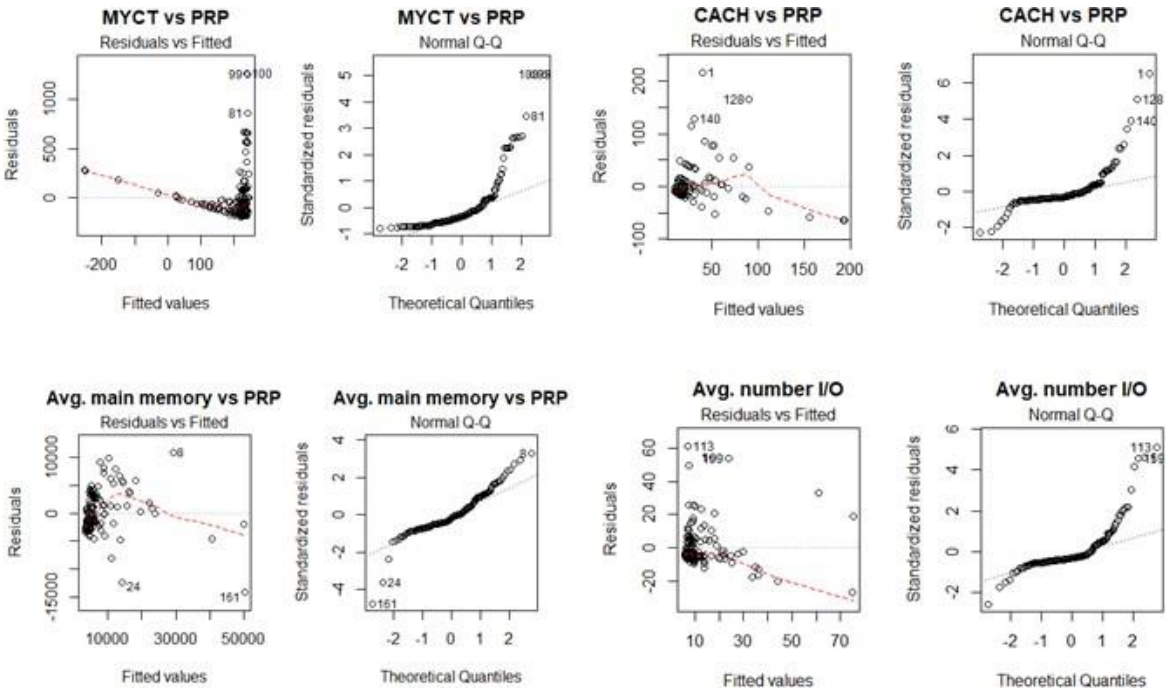

Figure 2: Residual Analysis for individual features against the target (PRP)

As the residual analysis shows a non-linear dependency for all involved features against our target, which can be seen as a deviation from the straight line $(x = y)$ in the Normal Q-Q plot. The residuals (left graph) support the impression of clustering and heteroskesdastic variances in our observations. This might be a sign that a classification algorithm will most likely be required to reach a better prediction result, given our data sample.

In the classification data, we make predictions based on the target EpiOrStroma, since this dataset contains 112 features and 600 data points, we will omit visualisations in favour of simplicity. However, the characteristics of this dataset are still highlighted in the classification section.

4. Regression Problem

In the regression task, the concepts of Random Forest Classifiers and Support Vector Regression are used. Surprisingly, the fitted data indicates that despite good predictive outcomes under SVR, Random Forest Classifiers outperformed these. Given that Random Forest Classifiers are more extensively used in classification contexts this result seems even more interesting. The high predictive accuracy of Random Forest Classifiers, in particular, made this algorithm the preferred choice in the regression framework.

Algorithm choice 1 - Support Vector Regression

As a subcategory of Support Vector Machines, SVR is most often applied to the context of linear regression models, where it is of interest to restrict a deviance margin (lower the cost) to the target variable y. Being a conceptual extension to the classical Least Squares estimate, SVR represents a convex optimization problem where underfitting or overfitting can be addressed by the inclusion of a regularization term such as found in Lasso or Ridge regressions. The general expression for SVR can be defined as follow.

Definition:

Given a training pair (x_i, y_i) where $x_i \in R^n$, $y_i \in R$ for $i = 1, \dots, l$. Then a linear Support Vector Regression finds a unique model w under which $w^T x_i$ is minimized in distance to the target value y_i.

It solves the following optimisation problem

$$\underbrace{Min}_{w} f(w) \quad \text{where} \quad f(w) = \frac{1}{2} w^T w + K \sum_{i=1}^{l} \varepsilon_\epsilon (w; x_i, y_i)$$

with $K > 0$ being a regularization parameter and,

$$\varepsilon_\epsilon(w; x_i, y_i) = \begin{cases} max(|w^T x_i - y_i| - \epsilon, 0) \\ max(|w^T x_i - y_i| - \epsilon, 0)^2 \end{cases} \text{ or}$$

Visually SVR is derived from SVM and can be depicted according to figure 3 below, where b is a marginal bias term that is often omitted in practical implementations.

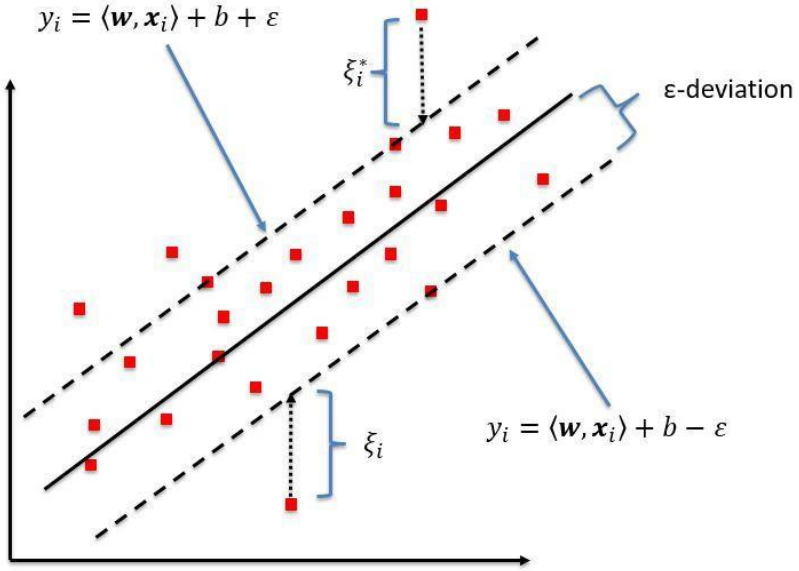

Figure 3: Visualisation of Support Vector Machines

Although clearly applicable to non-linear cases, in this particular example SVR delivered best results under linearity consistent with a hyperplane. Since the number of observations in the datasets is small, complexity issues based on primal input dimension are low as opposed to higher dimensional cases (such as Polynomial Kernels or RBF Kernels), which require a high number of support vectors.

Apart from being computationally simpler, linearity conditions under SVR have proven to be advantageous in effectively generalizing unseen data. Since SVR only requires a subset of the training data, learning rates were generally consistent throughout the model runs.

Algorithm choice 2 – Random Forest Classifier

Random Forest classifier has the CART functionality, which differentiates between classification and regression problems. Usually, in regression, RF has to fit a model on the dataset to test it against a real value. Several combinations of models are fitted by including the dataset variables based on the relation it has with the predicted values. The random forest adds new instability to the variables selected. It usually creates trees with 2/3rd of variables of the dataset. The accuracy of the random forest regression depends on the random state. In the metrics presented here, different states were used to understand their relationship with the model.

To improve random forest, cross-validation is used by splitting the data into train and test samples and then fit the model against them. It helps to understand the error and accuracy scores of the model in order to obtain an objective performance measurement. The regression tree is hard to interpret, as there would be several combinations of trees. However, different metrics help to understand which variables are to include in the regression model. Since the dataset is very small, it was attempted to fit a model with all variables included.

Performance Analysis and Metrics

Despite achieving accurate results under SVR, it was observed that the Random Forest Classifier is superior in almost all runs. This is perhaps surprising for the fact that RFC is a tool mainly used for classification rather than regression contexts. However, the data appears to be clustered and certainly not normally distributed. This is an important factor in justifying the strength of RFC.

Table 1: Performance comparison between the SVR and RFC algorithms

Algorithms	R-squared Score (X_train)	R-squared Score (X_test)
Support Vector Regression (SVR)	0.88095	0.71428
Random Forest Classifier (RFC)	0.88690	0.85714

As the above table indicates, the R-squared values achieved by the RFC-Algorithm outperform those obtained by SVR. Thus, RFC explains more variation in the data. It is worth mentioning that with different choices of random state values, the accuracy scores

fluctuate without any specific pattern. However, in some cases, there is a negative relationship between the accuracy level and error score, so it represents a trade-off mechanism.

5. Classification Problem

Despite having achieved a somewhat satisfactory result in the prediction using regression algorithms such as SVR, it is also possible to prove that classification generally does a better job in the given context. In the next section, the success rate between the Random Forest Classifier and the Extra-Trees Classifier will be compared. Finally, it will be decided which of the two algorithms should be implemented to obtain more accurate classification results.

Algorithm 1 – Random Forest Classifier

Random Forest is an appropriate algorithm to fit the model at an early stage of prediction as it gives a better understanding of the data structure. The decision trees are based on certain rules to split the trees. Overfitting the model is always a concern when using this classifier, however, with a lower number of n estimators the risk of overfitting reduces, as will be shown in this analysis. Random forest handles both missing data and categorical values.

Algorithm step for the top-down approach

The main idea of the top-down approach for random forests is to divide the problem into subproblems and then solve each problem individually. Such an algorithm generally works in the following sequence.

1. Randomly select m subsets from the dataset d, where each subset is smaller than the original dataset, that is $m_i < d$ for $i = 1, \dots, m$ where $m = \sum_{i=1}^{m} m_i$.
2. Split observations (instances) into m subsets (features) and separately find the best split from the node for each tree branch.
3. Using recursion for the s splits on all m subsets, select only instances reaching the same tree branch.
4. Stop recursion procedure if all instances have the same class within a branch.
5. If realisations in s belong to sthe ame class, return new leaf c.
6. Chose a final class for (stable) prediction over majority voting.

Below is a simplified representation of a Random Forest with n trees and chosen instances in each branch sharing the same class (orange colour).

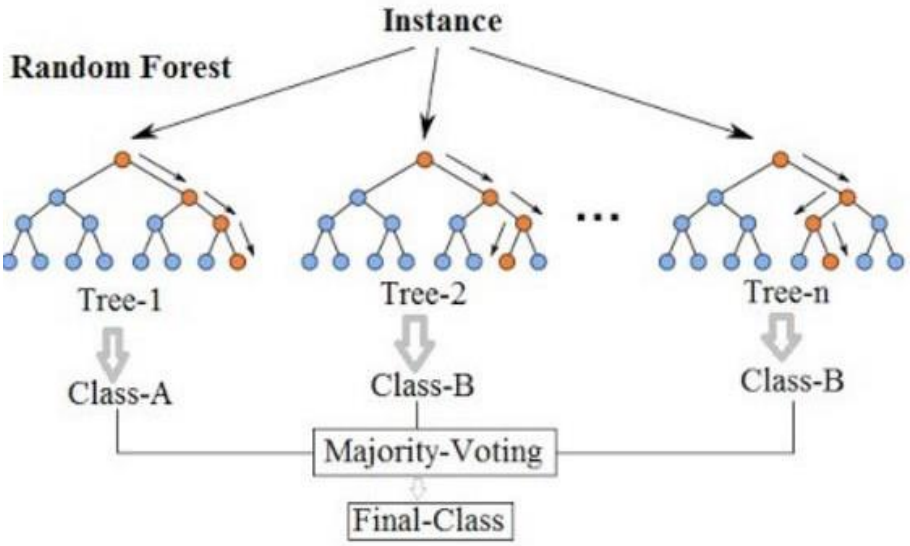

Figure 4: Class selection procedure for final prediction

Algorithm 2 – Extra Trees Classifier

Extra Trees classifier (Extremely Randomized Trees) is an alternative form of a random forest algorithm where at each step the entire dataset is used and decision boundaries are picked up at random. It always computes the splits and thresholds at random for each feature and the best of the generated thresholds are used for splitting the trees. This algorithm allows to reduce the variance of the model compared to other trees and also reduces computational burdens.

The main objective of extra trees is to further randomize tree building in the context of numerical input features, where the choice of the optimal cut-point is responsible for a large proportion of the variance of the induced tree. It often provides accuracy because of its smoothing. Extra trees are generally cheaper to train from a computational point of view but can grow much larger. Sometimes it can generalize better than Random Forests.

The ExtraTreesClassifier (RFE) class implements an estimator that fits a number of randomized decision trees on various sub-samples of the dataset and uses averaging to improve the predictive accuracy and control over-fitting. We have parameters n_estimators (the number of trees in the forest), max_features, bootstrap and also random_state and we calculate the score of the classifier of the model using cross_val_score.

Algorithm steps

Split a node (Dataset)

In: *Dataset*

Out: I*f split* $[n < n_c]$ *do nothing, where* n_c *is* a *random cut.*

– *If stop split*(*dataset*) *is TRUE then return nothing*

– *Else select* K *attributes* $\{n_1, \dots, n_K\}$ *among the dataset*

– Draw K splits $\{d_1, \dots, d_K\}$, *where split = Pick random_split*(*dataset,* n_i) for $i = 1, \dots, K$

– *Return a split and max of score* (*dataset, split*)

Random_split (Dataset, n)

In: *Subset S and attribute n*

Out: *n split*

– Let n_S^{min} *and* n_S^{max} be points based on values of S

– *Draw a random cut*

– Point n_c uniformly in [n_S^{min} , n_S^{max}];

– *Return the split* $[n < nc]$

Stop split (Dataset)

In: *Subset S*

Out: *Boolean expression*

– *If* $|S| < n_{min}$ *or constant output or attributes, then return TRUE*;

– *Else, return FALSE.*

Performance Analysis and Metrics

For the classification problem, significant performance differences between the two implemented Algorithms was observed as shown in table 2 below. The accuracy score and the mean squared error are chosen as metrics to determine the performance of the prediction. The outcome after running the Random Forest Classifier and Extra Trees algorithms on the dataset is also depicted in the table below.

Table 2: Performance comparison between the RFC and RFE algorithms

Algorithms	n estimators	Radom state	Accuracy	MSE
Random Forest Classifier (RFC)	1	22	0.95	0.2236
	1	10	0.938	0.2483
	20	10	0.995	0.0707
	8	8	0.99	0.1
Extra Trees Classifier (RFE)	-	10	1.0	0.0

As the above summary suggests, there is a clear relationship between the number of trees in the forest and the accuracy as well as mean squared error. It is also obvious that the number of random states is not influential for prediction. Thus, the training data was over-fitted. Using RFE is, therefore, not recommendable since the number of splits is highly randomized, which gives less control over the fitting procedure. With the RFC algorithm, it is possible to control some of the variation while still achieving plausible predictive results in the model fit (accuracy does not equal 1 and MSE different from 0).

6. Conclusion and choice of algorithm

Based on the overall accuracy and Kaggle submission scores, it was finally possible to restrict the choice in favour of the Random Forest Algorithm (RFC) for both the regression and the classification problems. This seems to be plausible in the light of the provided context, as the variables in the datasets display evidence of nonlinearity (inverse relationship) and appear clustered to some extent. The RFC captures the structure of the provided datasets more accurate when compared to SVR or RFE, which are less effective when dealing with clustered data. As the dataset was relatively small, there was no need to eliminate specific variables since it would not have improved the impact on the predictive accuracy.