**4. Learning Algorithms**

Machine learning is characterised by algorithms that can improve their ability to reason about a given phenomenon given greater observation and/or interaction with said phenomenon. Mitchell provides a formal definition of machine learning in operational terms: ``A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P if its performance at tasks in T, as measured by P, improves with experience E." \cite{mitchell1997}. Machine learning algorithms can be classified based on the nature of the feedback available to them: supervised learning, where the computer is presented with example inputs and desired outputs; unsupervised learning, where no labels are provided and the computer must find structure in its input; and reinforcement learning, where a computer interacts with a dynamic environment to perform a certain goal. These algorithms can be further categorised by desired output: classification, supervised learning that divides inputs into two or more classes; regression, supervised learning that maps inputs to a continuous output space; and clustering, unsupervised learning that divides inputs into two or more classes (basically, unsupervised classification). In this section, we describe the characteristics of the startup investment prediction task, review common machine learning algorithms, and ultimately determine which algorithms are most likely to suit the characteristics of this task.

**4.1. Task Properties**

Machine learning tasks are diverse and can be approached in many ways. For the current study, we will manipulate and combine the data collected from our data sources into a labelled data set appropriate for the application of supervised machine learning algorithms. Primary labels will be whether a startup received funding at each funding round, though measures of startup performance may also be investigated (e. survival time, exit). The key objective of machine learning algorithm selection is to find algorithms that make assumptions that are consistent with the structure of the problem (e.g. tolerance to missing values, mixed feature types, imbalanced classes) and suit the constraints of the desired solution (e.g. time available, incremental learning, interpretability). In Table 5 we have outlined some of the general properties of supervised learning tasks and identified the properties that are relevant to the current startup investment prediction task.

|  |  |
| --- | --- |
| **Properties** | **Relevance** |
| ***Problem*** |  |
| Missing values | 1 |
| Mixed feature types | 1 |
| Irrelevant features | 1 |
| Imbalanced classes (>10:1) | 1 |
| Small training set (< 100k) | 0 |
| High dimensionality (> 1k) | 0 |
| ***Desired Solution*** |  |
| Predictive power | 1 |
| Interpretability | 1 |
| Incremental learning | 1 |
| Ease of tuning | 0 |
| Computational speed | 0 |
|  |  |
| **Table 5.** Overview of the properties of the startup investment prediction task | |

**4.1.1 Problem**

**Missing values.** Data sets often have missing values, where no data is stored for a feature of an observation. Missing data can occur because of nonresponse or due to errors in data collection or processing. Missing data has different effects depending on its distribution through the data set. Public data sets, like CrunchBase and social networks, are typically sparse with missing entries despite their scale.

**Mixed feature types.** Data sets can contain data with distinct primitive natures: real-valued, interval, counts, rank, binary, ordinal, categorical and multi-categorical types. The simplest way to handle mixed data types is to convert into a unified type (e.g. real-valued, binary). However, this process partially destroys type-specific information. We expect mixed-feature types in our dataset as we will be handling data from databases, social networks and semantic text analysis.

**Irrelevant features.** Despite best efforts to only include features that have some theoretical relevance, most machine learning tasks will include irrelevant features. Irrelevant features are those that have no underlying relationship with classification. However, depending on the way they are handled they may affect classification or slow down the learning algorithm. We expect irrelevant features in our dataset because our proposed framework includes features that haven’t been tested in the literature.

**Imbalanced classes.** Data sets are not usually restricted to containing equal proportions of different classes. Significantly imbalanced classes are problematic for some classifiers. In the worst case, a learning algorithm could simply classify every example as the majority class. Our dataset is not dramatically imbalanced overall, but when looking at funding status for different funding rounds it is significantly imbalanced.

**Small training set.** Machine learning techniques generally work better with more data \cite{caruana2008}. Problems of small-data are numerous, but mainly revolve around high variance: over-fitting is harder to avoid, noise is an issue and outliers become more significant. Some machine learning techniques are better at dealing with these problems with others. Our dataset is reasonably large (500,000 startups and 150,000 funding rounds) so this is not a relevant property.

**High dimensionality.** In extremely high dimensional data sets, the number of features is larger than the number of observations. When dimensionality increases to the extent, the volume of the feature space increases so fast that the available data becomes sparse. This phenomenon is known as the curse of dimensionality or ``Hughes effect” \cite{hughes1968}. We do not expect our data set to be highly dimensional and where possible will transform the data into concise features.

**4.1.2 Desired Solution**

**Predictive power.** Predictive power is the ability of a machine learning algorithm to correctly classify new observations. If a model has no predictive power, then fundamentally the model is not representing the underlying statistical process being studied. For this reason, predictive power is a dominant property. However, predictive power is far from the only important property in machine learning selection. If algorithms provide similar predictive power, then other selection criteria become more significant.

**Interpretability.** Interpretability is the extent to which the reasoning of a model can be communicated to the end-user. There is a trade-off between model complexity and model interpretability. Some models are a ``black box” in the sense that data comes in and out but the model cannot be interpreted. For this study, it is important that we understand the determinants of the model so we are seeking algorithms that are highly interpretable.

**Incremental learning.** Incremental learning is where learning occurs dynamically whenever new observations are made and the algorithm adjusts what has been learned per the new observations. The key driver behind the need for incremental learning is when the underlying source generating the data is changing. It’s plausible that, as a system, the drivers behind startup investment are changing over time.

**Ease of tuning.** Machine learning algorithms have hyperparameters that must be tuned to ensure the model does not overfit its dataset. Some algorithms have many hyperparameters and tuning can be a computationally expensive process. We are not under significant time-pressure for this study and our dataset is not prohibitively large so this property is not especially relevant.

**Computational speed.** The amount of time and computational resources necessary to train a model varies a great deal between algorithms. Training time is often closely tied to predictive power. In addition, some algorithms are more sensitive to the number of data points than others. When time is limited it can drive the choice of algorithm, especially when the data set is large. For this task, with a relatively moderate data set and without significant time-pressure, this property is not especially relevant.

**4.2 Algorithm Properties**

Supervised machine learning are algorithms that can reason about observations to produce general hypotheses, and then make predictions about future observations. Supervised machine learning algorithms are diverse, from symbolic (Decision Trees, Random Forests) to statistical (Logistic Regression, Naïve Bayes, Support Vector Machines), instance-based (K-Nearest Neighbours), and perceptron-based (Artificial Neural Networks). The meta-review in Table 6 compares common supervised learning algorithms across general characteristics of the two domains mentioned in the previous section: assumptions about the structure of the problem and constraints of the desired solution. The following sections describe each candidate learning algorithm, critique their advantages and disadvantages, and present evidence of their effectiveness in relevant applications.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **Algorithms** | | | | | | | | | | | | | |
| **Properties** | **NB** | | **LR** | | **KNN** | | **DT** | | **RF** | | **SVM** | | **ANN** | |
| ***Problem*** |  | |  | |  | |  | |  | |  | |  | |
| Missing values | 2 | [3] | 1 | - | 0 | [3] | 2 | [3] | 2 | - | 1 | [3] | 0 | [3] |
| Mixed feature types | 2 | [3] | 2 | - | 2 | [3] | 2 | [3] | 2 | - | 1 | [3] | 1 | [3] |
| Irrelevant features | 0 | [3] | 0 | [4] | 1 | [3] | 2 | [3] | 2 | - | 0 | [3] | 0 | [3] |
| Imbalanced classes | 2 | - | 2 | - | 0 | - | 0 | [5] | 1 | - | 2 | [5] | 1 | [5] |
| Small training set | 1 | [2] | 2 | - | 2 | [2] | 2 | - | 2 | - | 2 | [2] | 0 | [3] |
| High dimensionality | 1 | [1] | 2 | [1] | 0 | [1] | 0 | [1] | 2 | [1] | 2 | [1] | 2 | [1] |
| ***Desired Solution*** |  | |  | |  | |  | |  | |  | |  | |
| Predictive power | 0 | [1] | 1 | [1] | 1 | [1] | 0 | [3] | 2 | [1] | 2 | [1] | 2 | [1] |
| Interpretability | 2 | [3] | 2 | [4] | 0 | [3] | 2 | [3] | 1 | [4] | 0 | [3] | 0 | [3] |
| Incremental learning | 2 | [3] | 2 | - | 2 | [3] | 1 | [3] | 1 | - | 1 | [3] | 2 | [3] |
| Ease of tuning | 2 | [3] | 2 | [4] | 2 | [3] | 2 | [3] | 1 | - | 0 | [3] | 0 | [3] |
| Computational speed | 2 | [3] | 2 | [1] | 1 | [3] | 2 | [3] | 1 | [1] | 0 | [3] | 0 | [3] |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| **Table 6.** Comparison of common supervised learning algorithms | | | | | | | | | | | |  |  |  |

**4.2.1 Naïve Bayes**

Naïve Bayes is a simple generative learning algorithm. It is a form of Bayesian Network that models features by generating a directed acylic graph, with the strong (naïve) assumption that all features are independent. While this assumption is generally not true, it simplifies estimation which means Naïve Bayes is more computationally efficient than other learning algorithms. Naïve Bayes can be a good choice for datasets with high dimensionality and sparsity as it estimates each feature independently. Naïve Bayes has been found to sometimes outperform more complex machine learning algorithms because it is reasonably robust to violations of feature independence, at least with regards to classification (Niculescu-Mizil & Caruana, 2005). However, Naïve Bayes is known to be a poor estimator of class probabilities, especially with highly correlated features (Zhang, 2004). Naïve Bayes was used alongside Logistic Regression, Decision Trees and Support Vector Machines to predict success in equity crowdfunding campaigns on the AngelList data set \cite{beckwith2016}. None of these models performed well. The algorithm that best predicted funded startups was Naïve Bayes with a Precision of .41 and Recall of .19, which means that only 19% of funded startups were classified correctly by the model. The author suggests the poor performance of their algorithms is caused by insufficient features captured in their training set, missing features relating to Intellectual Capital, 3rd Party Validation or Historical Performance. These features are included in the theoretical framework proposed by the current study.

**4.2.2 Logistic Regression**

Regression is a class of statistical methods that investigates the relationship between a dependent variable and a set of independent variables. Logistic regression is regression where the dependent variable is discrete. Like linear regression, logistic regression optimises an equation that multiplies each input by a coefficient, sums them up, and adds a constant. However, before this optimisation takes place the dependent variable is transformed by the log of the odds ratio for each observation, creating a real continuous dependent variable on a logistic distribution. A strength of Logistic Regression is that it is trivial to adjust classification thresholds depending on the problem (e.g. in spam detection (Hastie et al., 2008), where it is important that specificity is high). It is also simple to update a Logistic Regression model using online gradient descent, when additional training data needs to be quickly incorporated into the model. Logistic Regression tends to underperform against more complex algorithms like Random Forest, Support Vector Machines and Artificial Neural Nets in higher dimensions \cite{caruana2008}. This underperformance is observed when Logistic Regression is applied to startup investment prediction tasks \cite{xiang2012, beckwith2016, bhat2011}. However, weaker predictive performance hasn’t prevented Logistic Regression from being commonly used. Its simplicity and ease-of-use means it is used more casually, often being used without justification or comparative evaluation of its use \cite{gimmon2010, huang2015}.

**4.2.3 K-Nearest Neighbours**

K-Nearest Neighbours is a common lazy learning algorithm. Lazy learning algorithms do not perform explicit generalisation, but compare new instances with instances from training stored in memory. K-Nearest Neighbours is based on the principle that the instances within a dataset will exist near other instances that have similar properties (Cover and Hart, 1967). K-Nearest Neighbours models depend on how the user defines distance between samples; Euclidean distance is a commonly used metric. K-Nearest Neighbour models are stable compared to other learning algorithms and suited to online learning because they can add a new instance or remove an old instance without re-calculating \cite{kotsiantis2007}. A shortcoming of K-Nearest Neighbour models is that they can be sensitive to the local structure of the data and they also have large in-memory storage requirements. K-Nearest Neighbours was compared to Artificial Neural Networks to predict firm bankruptcy (Ahn & Kim, 2008). K-Nearest Neighbours is attractive in bankruptcy prediction because it can be updated in real-time. By optimising feature weighting and instance selection, the authors managed to improve the K-Nearest Neighbours algorithm to the point where it outperformed the Artificial Neural Network.

**4.2.4 Decision Trees**

Decision Trees use recursive partitioning algorithms to classify instances. Each node in a Decision Tree represents a feature in an instance to be classified, and each branch represents a value that the node can assume. Methods for finding the features that best divide the training data include Information Gain (Hunt et al., 1966) and Gini Index (Breiman et al., 1984). Decision Trees are close to an ``off-the-shelf” learning algorithm. They require little pre-processing and tuning, are interpretable to laypeople, are quick, handle feature interactions and are non-parametric. However, Decision Trees are prone to overfitting and have poor predictive power \cite{caruana2006}. These shortcomings have been addressed with pruning mechanisms and ensemble methods like Random Forests, respectively. Decision Trees were compared with Naïve Bayes and Support Vector Machines to predict investor-startup funding pairs using CrunchBase social network data \cite{liang2015}. Decision Trees had the highest classification accuracy and the authors suggest they are particularly useful in this application because their reasoning is easily communicated to startups.

**4.2.5 Random Forests**

Random Forests are an ensemble learning technique that constructs multiple Decision Trees from bootstrapped samples of the training data, using random feature selection \cite{breiman2001}. Prediction is made by aggregating the predictions of the ensemble. The rationale is that while each Decision Tree in a Random Forest may be biased, when aggregated they produce a model that is robust against over-fitting. Random Forests exhibit a performance improvement over a single Decision Tree classifier and are among the most accurate learning algorithms \cite{caruana2006}. However, Random Forests are more complex than Decision Trees, taking longer to create predictions and producing less interpretable output. Random Forests were used to predict private company exits using quantitative data from ThomsonOne \cite{bhat2011}. Random Forests outperformed Logistic Regression, Support Vector Machines and Artificial Neural Networks. This may be because the data set was highly sparse, and Random Forests are known to perform well on sparse data sets \cite{breiman2001}.

**4.2.6 Support Vector Machines**

Support Vector Machines are a family of classifiers that seek to produce a hyperplane that gives the largest minimum distance (margin) between classes. The key to the effectiveness of Support Vector Machines are kernel functions. Kernel functions transform the training data to a high-dimensional space to improve its resemblance to a linearly separable set of data. Support Vector Machines are attractive for many reasons. They have typically high accuracy \cite{caruana2006}, theoretical guarantees on limiting overfitting, and with an appropriate kernel they can work well even if data isn’t linearly separable in the base feature space (though this is an issue with a linear kernel). Support Vector Machines are computationally intensive and relatively complicated to tune effectively (compared to Random Forests, for example). Support Vector Machines were compared with back propagated Artificial Neural Networks in predicting the bankruptcy of firms using data provided by Korea Credit Guarantee Fund \cite{shin2005}. Support Vector Machines were found to outperform Artificial Neural Networks at this task, especially because it was on a small data set.

**4.2.7 Artificial Neural Networks**

Artificial Neural Networks are a computational approach based on a network of neural units (neurons) that loosely models the way that the brain solves problems. An Artificial Neural Network is broadly defined by three parameters: the interconnection pattern between the different layers of neurons, the learning process for updating the weights of the interconnections, and the activation function that converts a neuron's weighted input to its output activation. A supervised learning process typically involves gradient descent with back-propagation. Gradient descent is an optimisation algorithm that updates the weights of the interconnections between the neurons with respect to the derivative of the cost function (the weighted difference between the desired output and the current output). Back-propagation is the technique used to determine what the gradient of the cost function is for the given weights, using the chain rule. Artificial Neural networks tend to be highly accurate but are very slow to train and require significantly more training data than other machine learning algorithms. Artificial Neural Networks are also a black box model so it is difficult to reason about their output in a way that can be effectively communicated. Artificial Neural Networks have been rarely applied to startup investment or startup performance prediction tasks, probably because research in this area has used relatively small and low dimensional data sets. As one author put it ``More complex classification algorithms—artificial neural networks, Restricted Bolzmann machines, for instance—could be tried on the data set, but marginal improvements would likely result.” \cite{beckwith2016}. However, the current study seeks to address both of those issues so Artificial Neural Networks may be more relevant.

**4.3 Algorithm evaluation**

We evaluated common supervised learning algorithms for their suitability to the current task, startup investment prediction. In Table 7 we produce a ranking based on cross-referencing the task properties with the properties of the common algorithms. While this gives us some directionality of fit, we hesitate to rule in or out algorithms purely based on this ranking. Algorithm selection is complex and preliminary empirical testing will provide clarity as to which algorithms should be used. In addition, larger training sets and good feature design tends to outweigh algorithm selection \cite{caruana2008}. With those concessions in mind, we continue to review our findings. Our findings suggest that we should expect Random Forests, Support Vector Machines and Artificial Neural Nets to produce the highest classification accuracies. An ensemble of these high-performing methods may also provide an accuracy improvement, though at the cost of computational speed and interpretability. Random Forests could be expected to slightly outperform the other two algorithms due to robustness to missing values and irrelevant features and native handling of discrete and categorical data. However, Random Forests are not highly interpretable so Decision Trees and Logistic Regression might be preferable for early, exploratory analysis of the dataset.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **Algorithms** | | | | | | | | | | | | | |
| **Relevant Criteria** | **NB** | | **LR** | | **KNN** | | **DT** | | **RF** | | **SVM** | | **ANN** | |
| ***Problem*** | **2** | | 4 | | 6 | | **2** | | **1** | | 5 | | 7 | |
| Missing values | 2 | [3] | 1 | - | 0 | [3] | 2 | [3] | 2 | - | 1 | [3] | 0 | [3] |
| Mixed feature types | 2 | [3] | 2 | - | 2 | [3] | 2 | [3] | 2 | - | 1 | [3] | 1 | [3] |
| Irrelevant features | 0 | [3] | 0 | [4] | 1 | [3] | 2 | [3] | 2 | - | 0 | [3] | 0 | [3] |
| Imbalanced classes | 2 | - | 2 | - | 0 | - | 0 | [3] | 1 | - | 2 | [3] | 1 | [3] |
| ***Solution*** | **2** | | **1** | | 5 | | 5 | | **2** | | 5 | | **2** | |
| Predictive power | 0 | [1] | 1 | [1] | 1 | [1] | 0 | [3] | 2 | [1] | 2 | [1] | 2 | [1] |
| Interpretability | 2 | [3] | 2 | [4] | 0 | [3] | 2 | [3] | 1 | [4] | 0 | [3] | 0 | [3] |
| Incremental learning | 2 | [3] | 2 | - | 2 | [3] | 1 | [3] | 1 | - | 1 | [3] | 2 | [3] |
| ***Overall*** | **2** | | **2** | | 6 | | 4 | | **1** | | 5 | | 6 | |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| **Table 7.** Evaluation of common learning algorithms with respect to the current startup investment prediction task | | | | | | | | | | | | | | | |