Marie Guertin (260870552)
Luka Loignon (260871296)
Christian Martel (260867191)

Mini-Project #1: Machine Learning 101

Group 13

COMP 551 - Applied Machine Learning
Presented to Prof. Siamak Ravanbakhsh
Departement of Computer Science

September 30, 2021
McGill University

Abstract

In this project, we investigate the performance of two machine learning models – k-nearest neighbours and decision trees – on two different classification tasks. First, we explore the task of predicting if the income of an adult exceeds $50K/yr from various factors, such as age, sex, occupation, education level, etc. Second, we examine a complex letter recognition task. The objective is to determine which one of the 26 uppercase letters in the English alphabet is depicted in each black-and-white rectangular pixel image of the test data set. In the case of the Adult data set, we found that the decision tree model slightly outperformed the K-nearest neighbours model in terms of accuracy, training and prediction time. As for the Letter Recognition data set, we found that the k-nearest neighbour regression approach achieved much better accuracy than the decision tree approach and was significantly faster to train.

1 Introduction

The goal of this project is to gain experience in deploying supervised machine learning techniques to solve two real-world data science problems. We work with two data sets from the UCI Machine Learning Repository: the Adult data set [1] and the Letter Recognition data set [2]. The Adult data set was extracted from the 1994 US Census database by Barry Becker. It contains several attributes of different individuals such as age, sex, occupation, education level, relationship status, etc. For this data set, we try to predict whether someone makes over $50K/yr. The Letter Recognition data set was created by David J. Slate in 1991. It contains black-and-white rectangular pixel images of the 26 uppercase letters in the English alphabet. The images are based on 20 different fonts and were randomly distorted such that each one is unique. We tackle the problem of identifying the letter depicted in each of these images.

For both data sets, We compare the performance of two classification models: k-nearest neighbours (KNN) and decision tree. We investigate different data preprocessing techniques as well as the effects of hyper-parameters and data set size on the overall model performance. We use the following python libraries to analysis, manipulate and visualize our data: Pandas [3], scikit-learn [4], Matplotlib [5], Numpy [6] and XGBoost [7].

We found that both algorithm did not perform equally on both data sets. Indeed, for the Adult data set, our results show that the decision tree approach yields an accuracy of 85.95% which is almost 2% more accurate than using a KNN model. The decision tree model was also 2.8% faster to train. Conversely, the KNN model greatly outperforms the decision tree model when applied to the Letter Recognition data set. Precisely, it is 9.43% more accurate and 75% faster to train. These results are similar to other experiments conducted in the past on these data sets. [1][8]

2 Datasets

2.1 Adult Data Set

The Adult Data Set contains 48,842 data points, of which two thirds are used to train the model and the remaining third is used for testing. Each records contains a label indicating whether someone has a salary of less than $50K/yr. 75.9% of records in the training set have the class label *<=50K*. The dataset contains 8 categorical features – work class, education, marital status, occupation, relationship, race, sex, and native country – and 6 numerical features – age, survey weight (fnlwgt), education number, capital gain, capital loss, and hours per week.

The first step to preprocessing our data was to make sure it was formatted correctly. That included removing the dots at the end of the string label of the test set, removing white space, and converting the age feature from string type to data type. We also made sure the train set and test set columns had the same data type. The education number feature was dropped since its directly correlated to

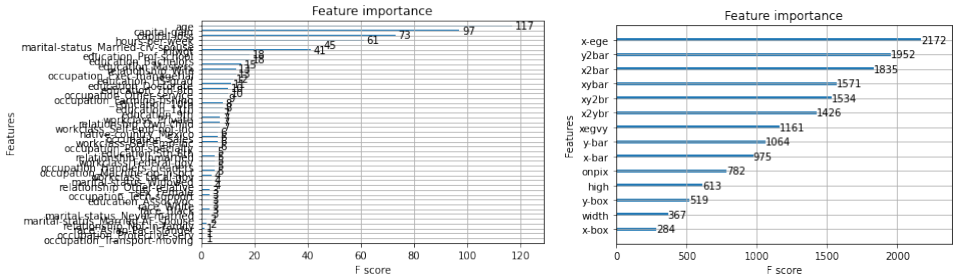

Figure 1: Adult (left) and Letter Recognition (right) Data Sets: XGBoost Feature Importance Plot

the education feature. Next, **data imputation** was performed on samples with missing or invalid values. Theses values were replaced with the mean value of the column in the case of continuous data and with the most occurring class for categorical data. The following steps were **one-hot-encoding** categorical features and **normalizing** all features. Finally, we used **gradient boosting** to determine the importance of each features and weighted the normalized data with each feature's F-score. As shown in Fig. 1, age, capital gain, capital loss, hours-per-week and married-civ-spouse (marital status) were the top five most important features in predicting the someone's salary.

2.2 Letter Recognition Data Set

The Letter Recognition Data Set contains 20,000 data points. We use the first 16,000 items for training and use the remaining 4,000 for testing. The data set contains 16 continuous features (statistical moments and edge counts extracted from each stimulus). Each item contains a label which corresponds to a letter of the English alphabet. Fortunately, this data set does not contain any missing data. Data from each column was normalized and once again, we used gradient boosting to determine which features were the most significant for the predictive task. As show in Fig. 1, x-edge, y2bar, x2bar, xybar and xy2br are the most important features. Each feature was weighted with their f-score such that more important features carry more weight in the prediction process.

3 Results

The results for each experiment are shown in Table 1. As expected, two different models don't perform equally on different data. For the Adult data set, the decision tree was the most accurate model and had the shortest training time. On the other hand, the KNN model was much more accurate than the decision tree model in the case of the Letter Recognition data set.

Table 1: Summary of results for each experiment

	Adult Test Set		Letter Recognition Test Set	
	KNN	Decision Tree	KNN	Decision Tree
K value	23	n/a	1	n/a
Maximum tree depth	n/a	10	n/a	29
Minimum samples/leaf	n/a	36	n/a	1
Accuracy (%)	84.135	85.953	93.475	84.050
Training Time (sec)	0.2199	0.2138	0.0658	0.1152

5-fold cross-validation was used to determine the optimal hyperparameters for each model. The results are also shown in Table 1. We found that changing the hyperparameters did not have the same effect on both data sets' performance (see Fig. 2 and Fig. 3). For example, when

implementing KNN on the Adult data set, increasing K up to 23 decreased the error rate on the validation set. Increasing K further did not affect the validation set's error rate anymore, but increased the error rate on the train set. Conversely, increasing the K value in the case of the Letter Recognition data set only increased the overall error rate. Therefore, 1 was the optimal K value for that data set. For the decision tree models, we decided to investigate the maximum tree depth and the minimum samples/leaf hyper-parameters. Since these two hyper-parameters are interdependent, we performed a grid search in order to find the parameter combination minimizing the error. Again, the effect of varying these parameters is very different on the two data sets (see Fig. 2 and Fig. 3). We found that a deeper tree with no minimum on the number of samples/leaf was preferable in the case of the Letter Recognition data set, while a smaller tree with a greater minimum of samples/leaf was best for modelling the Adult data set. We also investigated the effect of reducing the amount of data on the results of all 4 experiments. As expected, we found that reducing the amount of data greatly diminished the precision of the error rate results and increased the standard deviation of the cross validation errors.

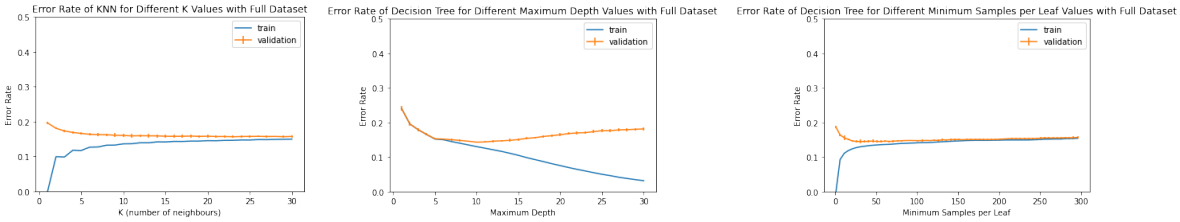

Figure 2: Adult - Hyperparameter tuning: K, maximum depth, minimum samples/leaf

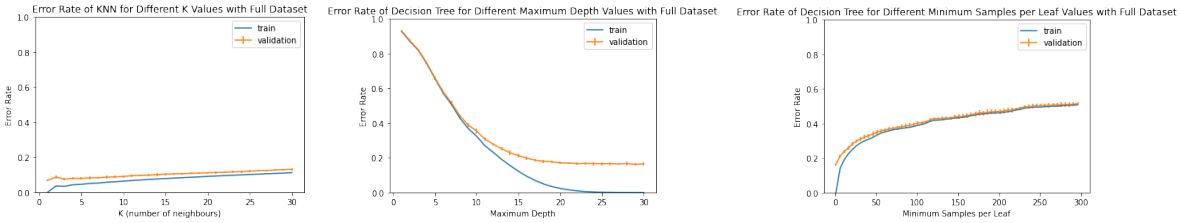

Figure 3: Letter Recognition - Hyperparameter tuning: K, maximum depth, minimum samples/leaf

4 Discussion and Conclusion

This project was a great first-hand experience with implementing machine learning models. We experimented with different preprocessing techniques such as one-hot-encoding, normalization and gradient boosting. We also observed that reducing the amount of data leads to poorer results and that tuning hyperparameters is essential to obtaining good results. Our results confirmed that different machine learning models perform differently on different data. Thus, model evaluation is a very important step when building a model to perform a predictive task. For future experiments, it would be interesting to see if we could achieve better results with a different model or if we could reduce training and prediction time without degrading our results.

5 Statement of Contributions

The workload was broken down equally across all team members as follows:

- Marie Guertin: data analysis, preprocessing, testing, debugging, report
- Luka Loignon: decision tree experiments, testing, debugging, report
- Christian Martel: project setup, preprocessing, knn experiments, testing, debugging, report

References

[1] Barry Becker. *Adult Data Set*. 1994. URL: https://archive.ics.uci.edu/ml/datasets/Adult.

[2] David J. Slate. *Letter Recognition Data Set*. 1991. URL: https://archive.ics.uci.edu/ml/datasets/Letter+Recognition.

[3] The pandas development team. *pandas-dev/pandas: Pandas*. Version latest. Feb. 2020. DOI: 10.5281/zenodo.3509134. URL: https://doi.org/10.5281/zenodo.3509134.

[4] F. Pedregosa et al. "Scikit-learn: Machine Learning in Python". In: *Journal of Machine Learning Research* 12 (2011), pp. 2825–2830.

[5] J. D. Hunter. "Matplotlib: A 2D graphics environment". In: *Computing in Science & Engineering* 9.3 (2007), pp. 90–95. DOI: 10.1109/MCSE.2007.55.

[6] Charles R. Harris et al. "Array programming with NumPy". In: *Nature* 585.7825 (Sept. 2020), pp. 357–362. DOI: 10.1038/s41586-020-2649-2. URL: https://doi.org/10.1038/s41586-020-2649-2.

[7] xgboost developers. 2021. URL: https://xgboost.readthedocs.io/en/latest/python/index.html.

[8] *Letter Recognition using kNN*. 2021. URL: http://amroamroamro.github.io/mexopencv/opencv/knn_ocr_letters_demo.html.