Comparison of Heterogeneous Bagging Ensemble using Neural Networks and Weak Learners and the impact of imbalanced data

School of Computing and Communications
MSc Data Science

***Abstract*—This report studies the performance of an ensemble that is consisted of three neural networks (NN) and three weak models. The weak models that are going to be used are a Decision tree, a Logistic Regression classifier and a Support Vector Classification (SVC) classifier. More than thirty thousand samples are used to create the two ensembles that can predict if the income of a person surpasses 50.000$ a year or not. The data set used in the training and testing processes is the Census Income Data Set, which contains many features for different people, like the Age, the Education level and the weekly working hours of each person. The results show that with the right pre-processing of the data, the neural networks ensemble model has a much better performance than the simple ensemble model. It is also shown that without properly balancing the data set, both models show very low performance. The overall performance will be judged by many metrics such as accuracy, but also precision, recall and more.**

I. Introduction

Ensemble learning is a well-known domain in the Machine Learning field. It is an approach to machine learning that tries to achieve performance, such as a lower error on regression or high accuracy for classification, by combining the predictions from multiple models. Today, ensemble learning has many real-world applications, including object detection and tracking, scene segmentation and analysis, image recognition, information retrieval, bioinformatics, data mining, etc.[1] All these applications, make the ability to accurately predict the category of a text very important. One of the most known problems that was solved using ensemble learning, is the face recognition of digital cameras. It was solved using a high-performance face detector based on boosting, one of the methods of ensemble learning.[1]

There are many methods of ensemble learning, with the major ones being Boosting, Bagging and Stacking. Boosting is an algorithm used for primarily reducing bias and variance. Boosting trains weak learners sequentially. After one weak model finishes training, it passes its results to the next weak learner and the same process is repeated. After parsing through all weak learners, the final model is converted into a strong learner. Bagging, also known as Bootstrap aggregation, is a machine learning ensemble method that aims to increase the accuracy and stability of machine learning algorithms used in statistical classification and regression. It also helps to avoid overfitting and significantly decreases the variance without increasing bias. Bagging works so well because of diversity in the training data since the sampling is done by bootstrapping. Finally, Stacking trains many different models in a complete data set, and then another model is trained on the outcomes of the all base-level model as a feature. Stacking improves the model prediction accuracy.[2]

This report also shows the significance of pre-processing the training data, specifically in this case balancing the imbalanced data. Fig.1 shows the distribution of the two classes in the data set used. As shown in Fig.1, an imbalanced data set refers to a data set where the target class has an uneven distribution of observations. In binary classification, this means that one class label has a very high number of observations and the other has a very low number of observations. An in-depth analysis is performed on the performance of both the ensembles when trained with an imbalanced data set and a balanced data set. As expected, properly balancing the data greatly affect both models' performance in a good way, with the most definite impact being on the precision recall and f1-score.

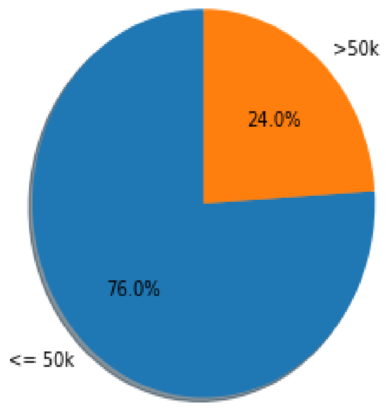

Fig. 1: Distribution of the two classes of the data set

In this report, an uncommon form of bagging called heterogeneous bagging is implemented, meaning that instead of training the same model with different bootstraps of the data, three different weak learners and three basic neural networks are trained. They are first trained using an imbalanced data set and then using the balanced form of the same data set. Then, their prediction is undergone the majority vote process. In this process, every individual classifier votes for a class, and the majority wins. In statistical terms, the predicted target label of the ensemble is the mode of the distribution of individually predicted labels. The data set used in this report is the Census Income Data Set, which was extracted by Barry Becker from the 1994 Census database.[4] After basic cleaning, it consists of thirty-two thousand samples of people. For each person, the data set contains fifteen features.

There has not been any significant study that compares Neural Networks and weak learners ensembles, which was the main motivation for this research.

II. Related Work

Several settings pose nontrivial challenges to machine learning algorithms, like class imbalance, concept drift and the Curse of dimensionality. Ensemble learning can be used to deal with those challenges, resulting in better results and the advance of machine learning as a whole.[5] This has motivated many researchers to try different variations of ensemble learning that can produce high-quality results.

Firstly, the ensemble approach for the classification of imbalanced data has been studied by (Nikulin & Ng, 2009)[6]. They suggested that a large number of relatively small and balanced subsets where representatives from the larger pattern are to be selected randomly should be considered, to produce a matrix of linear regression coefficients. After that, they evaluate the stability of the influence of each feature and only include the ones with stable influence for the model training. It is important to note that the proposed method is general and may be implemented using many different weak base learners. Although there was a significant improvement in the AUC score, compared to before applying their method, the results were still not satisfying, with their test result being AUC = 0.7023. So, even if this method improves the results using imbalanced data, it does not yet surpass the results received using balanced data.

An attempt to use the Heterogeneous Bagging Ensemble was made in 2010 by Qiang Li Zhao, Yan Huang Jiang & Ming Xu[7]. In their research, they analyse the advantages and disadvantages of Learn++, which is derived from the famous ensemble algorithm, AdaBoost[9]. They believe that Learn++ can potentially support heterogeneous base classifiers. After that, they introduce a new method Bagging++, which is based on the famous ensemble method of Bagging. The results show that heterogeneous Bagging++ can generalise better and learn faster than other similar methods such as Learn++ and NCL. One downside of their research is that the algorithm they used for the creation of the heterogeneous ensemble is not very effective, since it takes much more time than the training of a homogeneous ensemble.

In 2020, there was an attempt from V.Sobanadevi & G.Ravi to tackle[8] the data imbalance issue in ensemble learning, for credit card fraud detection. After pre-processing the data, they used bootstraps of the data set to train different models. After that, they use the aggregated predictions as an input for a meta-learner for Stacking, which provides them with the final predictions. The proposed model handles imbalance effectively, and the heterogeneity of the models provides a good solution for the noisy data. Compared to existing state-of-the-art models for fraud detection, the proposed model provided fewer false negative rates. Therefore it ensures high performance and efficiency. The only drawback is that the proposed model can mainly be applied in the financial domain in real time transactions, so for the most part it can't be generalised.

Neural Networks have been used to form ensemble models for more than ten years. From as early as 1990, Lars Kai Hansen & Peter Salamon[10] introduce the idea of using not a single neural network but an ensemble of neural networks, each of which has been trained on the same database. After that, they would obtain a classification value from each neural network and then use a consensus scheme to decide the collective classification by vote. They proved that using an ensemble of neural networks with a plurality consensus scheme can be expected to perform far better than using a single neural network. The results could be further improved, by further fine-tuning the neural networks, and training them for different samples of the given data set which is the concept of Bootstrapping.

Although much work has been done in the field of ensemble learning, both using neural networks and weak learners, no extensive comparison of those two methods has been done. Some comparison studies have been done comparing ensemble classifiers with single deep neural networks, like Shahab Jozdani, Brian Johnson & Dongmei Chen[11], but they still don't research neural network ensembles compared to weak learner ensembles.

III. Methodology

A. Data Sourcing

The data used in this experiment is the Census Income Data Set, and Google Colab was used as the platform for the algorithm creation. To import the data in Colab, the data was uploaded to Google Drive and using the 'drive' library from Google Colab, the drive was mounted to the Colab file. After that, the pandas library was used to load the data from the drive.

B. Data Pre-Processing

In this project, the models are trained in two cases. In the first one, the data is left imbalanced and in the second one, the data is balanced. The pre-processing process differs between the two cases, but some of the steps are common. The pre-processing for both cases is described below.

The first step is to search the data for NULL values. The first thing to do is to use the pandas library and see the info of all the features. In this case, there were no NULL values detected. But upon further examination of the data, many cases where instead of a normal value the value ' ?' were spotted. It was concluded that instead of normal NULL values, the specific data set chose the value ' ?' as its NULL value, so they had to be removed. To do that, all rows that had one or more features with the ' ?' value were dropped from the data set. This was a common process in both cases.

During the pre-processing, it is vital to take the data types of all the features into account. In this data set, nine of the data features had type 'string', and four of the data features had type 'int'. For both the test cases, there was a need to encode the string values into numerical ones, so they can be used as input for the training of the models. For that purpose, the LabelEncoder module of the sklearn library was used. Using this, nine encoders were created (one for each string feature) and trained. For the training, each encoder used a different feature as training data. Finally, all the string data was transformed, using the nine encoders, into numerical values representing their actual value. This was also a common process in both cases.

After that, the balance of the data set was checked. After counting the values of the target variable, it was shown that there were 22654 samples of people with an income '<=50k' and 7508 samples of people with an income '>50k'. This translated to one class expressing approximately 75% of the total data set, while the other class expressing just the remaining 25%. This shows a major imbalance in the data set. Since one of the purposes of this project is to review the impact of balancing the data set used in the Heterogeneous Bagging Ensemble, the data set that was processed up to this point is used to train a Bagging Ensemble composed of three Neural Networks, and a Bagging Ensemble composed of three weak learners, to compare those results with the results taken using the balanced data set that is going to be composed.

Since the data for the first case is ready, the data for the second case (balanced data) must be prepared. The data imbalance problem has many different solutions. The most common ones are undersampling and oversampling. Oversampling and undersampling are opposite and roughly equivalent techniques. They both involve using a bias to select more samples from one class than from another. Undersampling methods work by reducing the majority class's samples. Some informed undersampling methods and iteration methods also apply data cleaning techniques to further refine the majority class samples. A drawback to undersampling is that the sample of the majority class chosen could be biased. To balance the data set, new samples are added to the minority class in the oversampling method. The two types of oversampling algorithms are random oversampling and synthetic oversampling. Existing minority samples are duplicated in the random oversampling technique to augment the size of a minority class. In the synthetic oversampling technique, artificial samples are generated for the minority class samples. These new samples provide critical information to the minority class, preventing misclassification of its samples.[12] The main disadvantage of oversampling is that by making exact copies of existing samples, it makes the model likely to overfit.

The method used in this project is the Synthetic Minority Oversampling Technique (SMOTE) method. Duplicating samples from the data set can balance the data, but does not provide any additional information to the model. So the best solution is to synthesise new samples from the minority class. SMOTE was developed in 2002 by Nitesh Chawla, et al.[13], and works by selecting examples that are close in the feature space, drawing a line between the examples in the feature space and drawing a new sample at a point along that line[14].

By using SMOTE with the 'minority' sampling strategy, new samples of the minority class are added to the previous data set until the two classes are perfectly balanced. After that, the new data set has 22654 samples for both classes, with the total number of samples being 45308.

Finally, the data set will undergo feature selection. The 'Capital_Gain' and 'Capital_Loss' features of the data set, are greatly imbalanced, with more than 90% of them being 0. Hence, they will not provide useful information for the model training and are discarded.

C. Model Creation

The sklearn library was used to create the three weak models (Decision tree, a Logistic Regression classifier and a Support Vector Classification (SVC)) using the default parameters, with the only change being the (probability=True) parameter, so access to the probabilities for each prediction can be granted for the calculation of some metrics for the evaluation.

The three neural networks were designed using the Keras module of the TensorFlow library. All three of them are sequential models with three layers. The first one is common to all three and is the input layer. The dimension of the input layer is the length of each model's training data set. The next layer is a Dense layer, which is a layer the neurons of which are connected to every neuron of its preceding layer. The Dense layer uses the same kernel_initializer 'he_normal', which means that the model draws samples from a truncated normal distribution centred on 0, for all the models. What differs between the three models, is the Dense layer's activation function, which decides whether a neuron should be activated or not. The first model uses the 'ReLU' activation function, which will output the input directly if it is positive, otherwise, it will output zero. The second model uses the 'LeakyReLU' activation function, which is a type of activation function based on a ReLU but has a small slope for negative values instead 0. The third model uses the 'sigmoid' activation function, also known as the logistic function, which passes the input through a mathematical formula (Equation 1) and returns a value between 0 and 1.

$$S(x) = \frac{1}{1 + e^{-x}} \tag{1}$$

Finally, the third layer is common for all three models. It is a single node Dense layer, which will contain the predicted value. The third layer uses the 'sigmoid' activation function for all models. The main reason behind this is that in binary classification problems, the output of a layer with the 'sigmoid' activation can be interpreted as confidence values that with the right processing can be used for the visualisation of the ROC graph. The architecture of the neural network models used for the ensemble is shown in Fig.2.

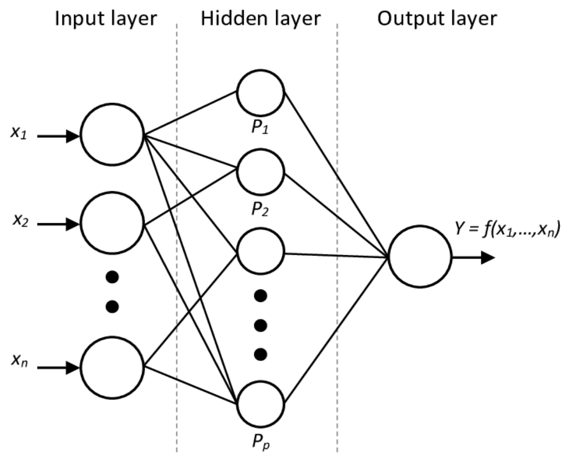

Fig. 2: Neural Networks architecture

D. Model Training

Before the training process, the data set has to be split into the training set and the test set. In both cases, for most of the models, the data is split randomly, with 80% of the data assigned to train the models and 20% of the data used to test their performance. But in the balanced data case, for the SVC model and the third of the neural networks, the data is split randomly, with 60% of the data assigned to train the models and 40% of the data used to test their performance. This happens because the SVC model will take much more time to train with a higher number of samples since the data was oversampled. Quoting the sklearn SVC website: 'The fit time scales at least quadratically with the number of samples and may be impractical beyond tens of thousands of samples'.

The weak classifiers are trained using the default parameters, each one with its bootstrap of the data set. The neural networks are all trained for 20 epochs, with all of them using the 'binary_crossentropy' loss function and the 'adam' optimizer, which is the most common and useful optimizer in most cases. The binary crossentropy loss function calculates the loss of an example by computing the following average[15]:

$$\text{Loss} = -\frac{1}{\substack{\text{output}\\\text{size}}} \sum_{i=1}^{\substack{\text{output}\\\text{size}}} y_i \cdot \log \hat{y}_i + (1 - y_i) \cdot \log(1 - \hat{y}_i) \quad (2)$$

where $\hat{y}_i$ is the i-th scalar value in the model output, y_i is the corresponding target value, and output size is the number of scalar values in the model output.

For each epoch of the training, the training error and accuracy are measured, to check the rate that they change. Finally, each model was tested using a test data set. The results of the experiments are shown in the next section.

IV. Experimental Results

After training all the models, it is time to test them and evaluate their performance. The models will be compared using the Precision, Recall, F1-score and accuracy, and for the ensembles using the balanced data AUC score and the ROC curve will be included.

* Accuracy: Accuracy can be defined as the percentage of correct predictions.

* Precision: Precision is the proportion of the positive identifications that were predicted correctly. Precision can be defined as:

$$\text{Precision} = \frac{TP}{TP + FP} \quad (3)$$

* Recall: Recall expresses the proportion of actual positives that were identified correctly. Recall can be defined as:

$$\text{Recall} = \frac{TP}{TP + FN} \quad (4)$$

*F1-score: F-score, also called the F1-score, is a measure of a model's accuracy on a data set. In cases where the classes are balanced and there is no major downside to predicting false negatives, accuracy is used, but when the data set is imbalanced and there are consequences in predicting False Negatives, F1-score is used. F1 score is a way of combining the precision and recall of the model, and it is defined as the harmonic mean of the model's precision and recall.[16] F1-score can be defined as:

$$F1 = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}} = \frac{TP}{TP + \frac{1}{2}(\text{FP} + \text{FN})} \quad (5)$$

AUC: The Area Under the Curve (AUC) measures the ability of a classifier to tell the classes apart. It can be calculated like:

$$AUC = 1/2 - \text{FPR}/2 + \text{TPR}/2 \quad (6)$$

ROC curve: The ROC curve is a probability curve that is plotted with TPR against the FPR, and is a visual representation of the separability of the classes.

In the above equations, TP refers to the number of cases where the positive class was correctly predicted, FP refers to the number of cases where the positive class was falsely predicted, FN refers to the number of cases where the negative class was falsely predicted and TPR and FPR refer to the percentage of TP and FP respectively.

A. *Imbalanced data*

First the results of each of the Heterogeneous Bagging ensembles using the imbalanced data are shown.

1) *Weak learners ensemble*

After training the ensemble's models and testing their accuracy, it is observed that the decision tree classifier overfits. This can be concluded by comparing its accuracy when predicting the test data and the data used to train it. For the test set, the decision Tree Accuracy is 0.82, while for the train set it is approximately 1.0. So this will also test if using the ensemble will deal with the overfitting problem. The classification report, which contains all the metrics for the model's performance is displayed in Table I.

Class	Precision	Recall	F1-score
0	0.81	0.99	0.89
1	0.93	0.28	0.43

TABLE I: Classification Report for weak learners ensemble

The accuracy of this model is 81.6% when using the test set, and 81.7% when using the training set as input. In Fig. 3, the confusion matrix of the weak learners ensemble results is displayed, where the class which was misclassified more becomes more apparent.

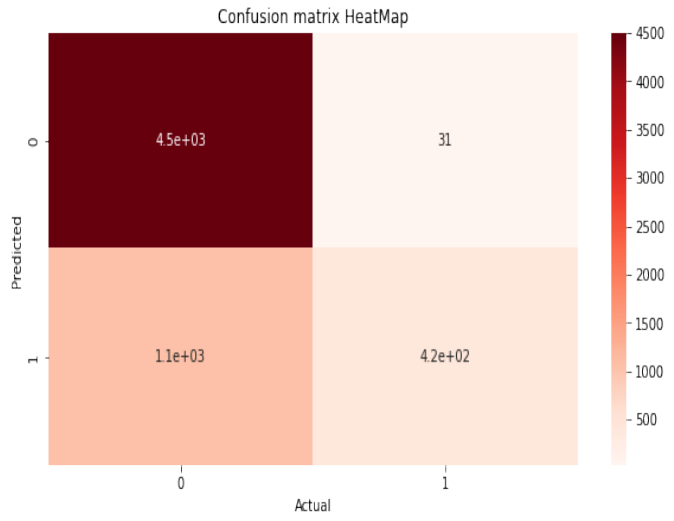

Fig. 3: Heat Map of Confusion matrix for weak learners ensemble

2) *Neural Networks ensemble*

After testing the neural networks against both the test data and the train data, they provided similar accuracy values, meaning that there is no sign of overfitting in this case.

Next are the results of the neural networks ensemble. The classification report can be seen in Table II.

The accuracy of this model is 81%. In Fig. 4, the confusion matrix of this model is displayed. It can be observed both here, but on the Fig.3 confusion matrix as well, that most of the model mistakes are concerning the class '0' (<=50k), which is understandable because the sample in those cases are greatly overfitted with class 0 having more than three times more samples than class 1 (>50k).

Class	Precision	Recall	F1-score
0	0.81	0.97	0.88
1	0.78	0.33	0.46

TABLE II: Classification Report for Neural Networks ensemble

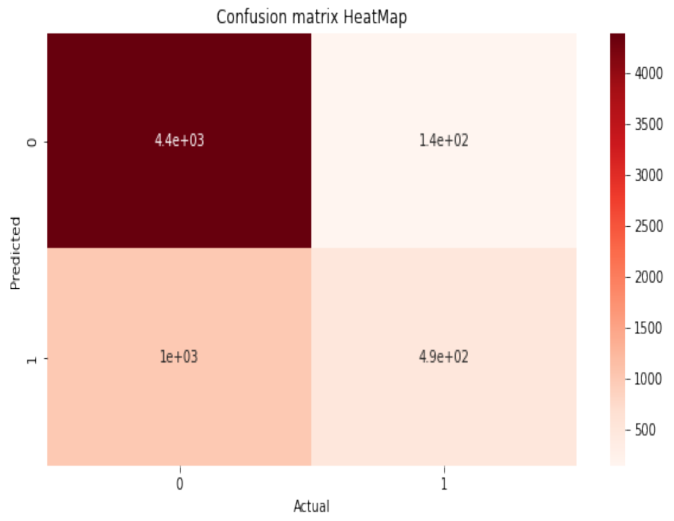

Fig. 4: Heat Map of Confusion matrix for Neural Network ensemble

B. *Balanced data*

Finally, the results of each of the Heterogeneous Bagging ensembles using the balanced data are presented.

1) *Weak learners ensemble*

Even after balancing the data, after testing the ensemble's trained models, the decision tree classifier seems to overfit the training data. This is shown by the classifier having an 83.3% accuracy when predicting the testing data and a 97.6% accuracy when predicting the training data. This model's precision, recall and f1-score are displayed in Table III.

Class	Precision	Recall	F1-score
0	0.83	0.74	0.78
1	0.77	0.84	0.80

TABLE III: Classification Report for weak learners ensemble

The accuracy of this model is 79.4% and its creation and training process took 187.7 seconds. In Fig. 5, the confusion matrix of this model is displayed.

The AUC value using the test data is 0.9. The ROC curve, showing the connection between TPR (True positive rate) or sensitivity and FPR (False positive rate) or 1-specificity, is displayed in Fig.6.

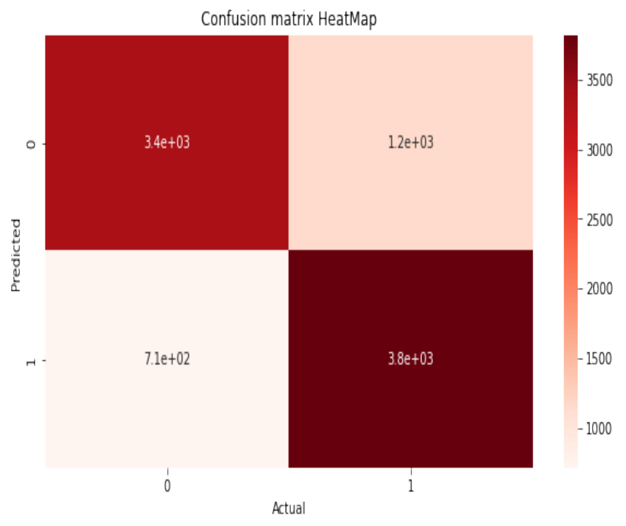

Fig. 5: Heat Map of Confusion matrix for weak learners ensemble (balanced data)

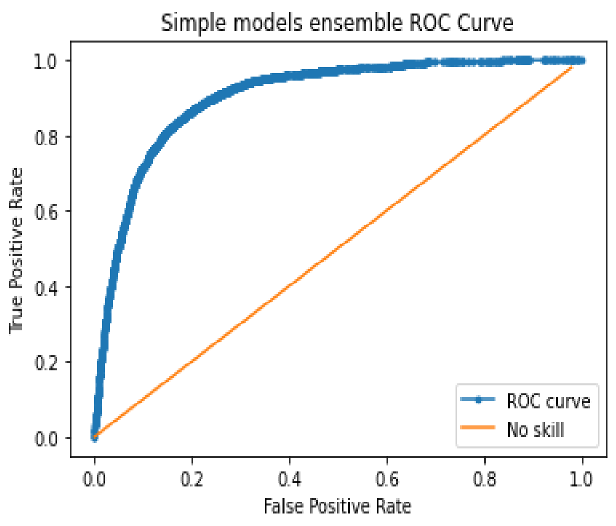

Fig. 6: ROC curve for weak learners ensemble (balanced data)

2) Neural Network ensemble

All three models that consist of the neural network ensemble were tested both with their test set and their training data set. None of the three showed any issue of overfitting, since their accuracy was similar in both cases. The model's classification report that shows the metrics of the model when tested with the same data set as the previous ensemble is shown in Table IV.

Class	Precision	Recall	F1-score
0	0.79	0.82	0.80
1	0.81	0.78	0.80

TABLE IV: Classification Report for neural network ensemble

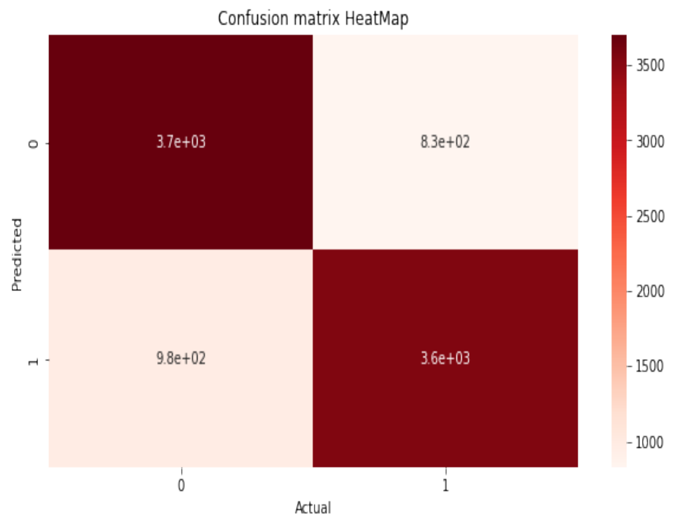

Fig. 7: Heat Map of Confusion matrix for Neural Network ensemble

The accuracy of this model is 80% and the model creation and training process took 367.3 seconds. The confusion matrix of the model is shown in Fig.7.

Finally, the AUC score that resulted from testing is 0.89, and the ROC curve of the model is shown in Fig.8.

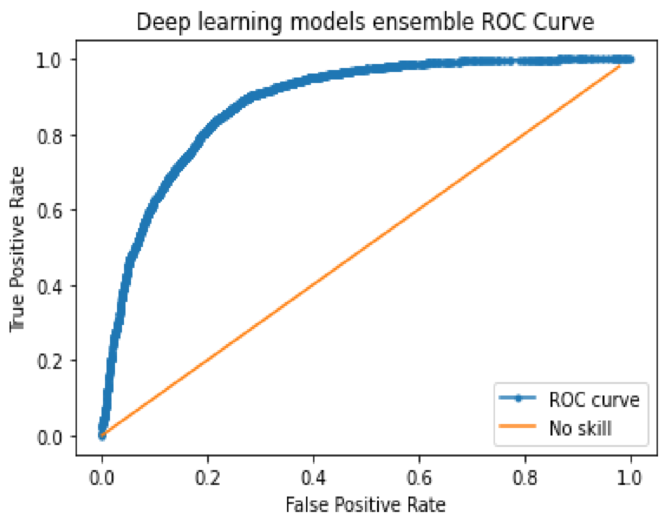

Fig. 8: Heat Map of Confusion matrix for Neural Network ensemble

All ensemble models were tested both with the training data and the testing data of all the individual models of which they were consisted to test for overfitting, but none of the ensemble models showed any problems in generalising their predictions.

V. Discussion

The first issue that this project wanted to address, is the importance of balancing the data used for model training. The two ensemble models were trained both in the case of imbalanced data, and after the data underwent some balancing techniques. Since class 0 (<=50k) was the dominating class, occupying 76% of the data, the results of the ensemble models that are trained using the imbalance data are logical and as expected. Both the weak learners ensemble and the neural networks ensemble have trouble predicting class 1 (>50k). The high precision and low recall in both cases show that the models are well trained to recognise class 0 cases, but have a very poor performance in predicting the class 1 cases. Both models display high accuracy, but that is irrelevant when dealing with a skewed data set since high accuracy can be achieved by always predicting the dominant class. So instead, the f1-score which accounts for the performance of all classes is used, which is again very low on both the models, meaning that either the models should be modified, or the data set should be balanced to improve data quality. After the data is balanced, it is clear from the higher values in both recall and f1-score, that both ensembles started to perform better, especially in predicting the class 1 cases. To be more precise they started giving more output-sensitive predictions, covering the False Negatives as well.

The second issue is the comparison of the ensemble classifier that consists of the three weak learners and the classifier that consists of three neural networks. The performance of the two models is very similar. The metrics of the classification report and the ROC curve show that they perform at the same level. The one that seems to perform just a bit better is the neural networks ensemble because it has higher f1-score values and while the weak learners ensemble has a little higher precision for class 0 and recall for class 1, the neural networks ensemble has a significantly higher recall for class 0 and a little higher precision for class 1. Their accuracy and AUC score values are almost the same, with the difference being non-significant. But even though the neural networks ensemble performs slightly better than the other, there is an issue with its training time. The process of the neural networks ensemble model creation and training takes twice the time of the weak learner ensemble. When including that variable in the mix, the weak learners ensemble is a better choice for data of this volume. This conclusion was not expected at the start of the project. The main reason that explains this outcome is the size of the data. Neural Networks are complex models in machine learning, that train thousands or millions of parameters to correctly classify a sample. As shown in Fig.9 in the appendix, shared by Sir Andrew Ng[17], simpler machine learning algorithms outperform or have the same performance as deep learning models up to a certain amount of data. After that, neural networks start outperforming all other models. So in theory, had the data set used been many times bigger, the neural networks ensemble model would clearly outperform the weak learners model.

VI. Conclusions and Future Work

In this report, two different ensemble models are created, trained and tested, using imbalanced and balanced data. The first model is created by combining three simple models, a Decision tree, a Logistic Regression classifier and a Support Vector Classification (SVC) classifier, while the second one is composed of three single layered neural networks, with some different parameters. The tests showed the negative impact of using imbalanced data on the models. But what is also shown, is that this problem can be fixed using different techniques, like SMOTE. For the comparison of the two models when trained with balanced data, it is shown that both models perform adequately, with the neural network ensemble being a bit better solution as far as results as concerned. But when the consumed time is factored in, the neural networks ensemble falls short of the weak models ensemble. So even though both models are acceptable, which one is preferred depends on how much time is available for the user. When bigger data sets are provided, the neural networks ensemble will most likely outperform the other model. It would be worth testing those two models in a clean big data set (hundreds of thousands of samples) to see at which point the neural networks ensemble starts to perform much better. It would also be interesting to add more different models in both ensembles and see how they perform.

Acknowledgements

The code for each model and the data needed to reproduce the results are all uploaded in Google Drive. You can gain access to them through the public share link here: SCC.462 Final Project.

References

[1] Cha Zhang and Yunqian Ma. 2012. Ensemble Machine Learning: Methods and Applications. Springer Publishing Company, Incorporated.

[2] Chauhan, A., 2021. ENSEMBLE METHODS—Bagging, Boosting, and Stacking. [online] Medium. Available at: <https://medium.com/analytics-vidhya/ensemble-methods-bagging-boosting-and-stacking-28d006708731>

[3] O'Reilly Online Learning. n.d. Machine Learning for OpenCV. [online] Available at: <https://learning.oreilly.com/library/view/machine-learning-for/9781783980284/47c32d8b-7b01-4696-8043-3f8472e3a447.xhtml>

[4] https://archive.ics.uci.edu/ml/datasets/census+income

[5] Sagi, O, Rokach, L. Ensemble learning: A survey. WIREs Data Mining Knowl Discov. 2018; 8:e1249. https://doi.org/10.1002/widm.1249

[6] Nikulin, V., McLachlan, G.J., Ng, S.K. (2009). Ensemble Approach for the Classification of Imbalanced Data. In: Nicholson, A., Li, X. (eds) AI 2009: Advances in Artificial Intelligence. AI 2009. Lecture Notes in Computer Science(), vol 5866. Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-642-10439-8_30

[7] Zhao, Q.L., Jiang, Y.H., Xu, M. (2010). Incremental Learning by Heterogeneous Bagging Ensemble. In: Cao, L., Zhong, J., Feng, Y. (eds) Advanced Data Mining and Applications. ADMA 2010. Lecture Notes in Computer Science(), vol 6441. Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-642-17313-4_1

[8] Sobanadevi, V., Ravi, G. (2021). Handling Data Imbalance Using a Heterogeneous Bagging-Based Stacked Ensemble (HBSE) for Credit Card Fraud Detection. In: Peter, J., Fernandes, S., Alavi, A. (eds) Intelligence in Big Data Technologies—Beyond the Hype. Advances in Intelligent Systems and Computing, vol 1167. Springer, Singapore. https://doi.org/10.1007/978-981-15-5285-4_51

[9] Freund and R.E.Schapire, A Decision-Theoretic Generalization of On-Line Learning and an Application to Boosting,J. Comput. Syst. Sci. (1997) no.1, 119-139 doi:10.1006/jcss.1997.1504

[10] L. K. Hansen and P. Salamon, ”Neural network ensembles,” in IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 12, no. 10, pp. 993-1001, Oct. 1990, doi: 10.1109/34.58871.

[11] Jozdani, Shahab & Johnson, Brian & Chen, Dongmei. (2019). Comparing Deep Neural Networks, Ensemble Classifiers, and Support Vector Machine Algorithms for Object-Based Urban Land Use/Land Cover Classification. Remote Sensing. 11. 10.3390/rs11141713.

[12] Shelke, Miss. Mayuri S., Dr. Prashant R. Deshmukh and Vijaya K. Shandilya. “A Review on Imbalanced Data Handling Using Undersampling and Oversampling Technique.” (2017).

[13] Kegelmeyer, W., Hall, L., Bowyer, K. and Chawla, N., 2011. SMOTE: Synthetic Minority Over-sampling Technique.

[14] Brownlee, J., 2021. SMOTE for Imbalanced Classification with Python. Machine Learning Mastery. Available at: <https://machinelearningmastery.com/smote-oversampling-for-imbalanced-classification/>.

[15] https://peltarion.com/knowledge-center/documentation/modeling-view/build-an-ai-model/loss-functions/binary-crossentropy

[16] Wood, T., 2021. F-Score. [online] DeepAI. Available at: <https://deepai.org/machine-learning-glossary-and-terms/f-score#: :text=The%20F%2Dscore%2C%20also%20called,positive%27 %20or%20%27negative%27.>

[17] Patil, R., 2021. Need of Deep Learning — Is there any need of Deep Learning?. [online] Analytics Vidhya. Available at: <https://www.analyticsvidhya.com/blog/2021/05/is-there-any-need-of-deep-learning/>.

Appendix

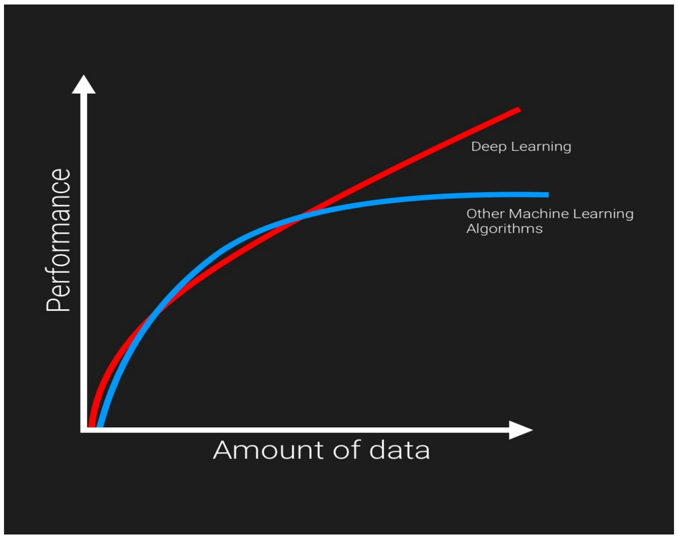

Fig. 9: Deep Learning performance based on the amount of data