Knowledge Distillation from Random Forests

(Supervised learning)

Mahrad Pisheh Var
Computer Science Department
University of Essex, Colchester
mpishe@essex.ac.uk

Github link : https://github.com/mpishe/Assignment2
Github link for the labs : https://github.com/mpishe/ce888labs

***Abstract*— Learning algorithms where purposed to structure a set of classifiers are called ensemble methods. The method is used to vote their predictions in the classification of the new data-points. In this project random forests will be used to create a cumbersome model. The model is quite computationally expensive therefore, a binning method is used to distillate the data. The new dataset is produced to be used on training a decision tree classifier. The new model will be experimented on three different datasets and their accuracies will be compared with state-of-art.**

Keywords—Data-science, Random Forests, Knowledge Distillation, Decision trees.

I. Introduction

A. Reasoning for using knowledge distillation

Artificial intelligence and data science have in recent years a high improvement rate. There has been built new models that allow the systems to perform better and more efficiently. The performance of these programs is measured by the accuracy of their output value. These advancements were made possible by breakthroughs that has occurred in the field of data science. One of the objectives for data science is to achieve high accuracy in decision making. This can be accomplished using uncomplicated techniques that improve performance of any machine learning algorithm. The algorithm will train different models on the same data and then average their predictions. Furthermore, this concept is undemanding as it can forecast using a full ensemble of models, however, it is quite cumbersome and can be computationally expensive. Random Forests - the ensemble method creates a large and complex model. [1] explains when a number of challenges occur when this concept was used in the practice. The ensemble method uses methods such as, boosting, bagging, random feature selection and stacking.

B. Purpose

When a model is produced by an ensemble method, the model will have high accuracy in predicting the labels, but it will take away the convenience in usability of the model. This paper will experiment with a cumbersome model created from a Random Forests and it will introduce a way to convert the model to be computationally reasonable and keep its high accuracy in predictions. Furthermore, this paper will also cover a whole background on every concept used in the project. All the concepts explained in the background will be experimented with on the chosen datasets. The methods accuracy will be calculated, and their performance will be visualised.

II. Background

A. Decision Tree Classifier

1) Decision Tree overview

A Decision tree is a classification method, it consists of a structure that can be used to simply explore Nodes. This is by satisfying conditions in each node. The structure of a Decision Tree consists of:

1- Root Node - this contains no incoming edges, but it has zero or more than one outgoing edges [10].
2- Internal Node – this has one incoming edge and two or more outgoing edges [10].
3- Leaf or terminal Node – this has no outgoing edges and one incoming edge; this Node holds the Class label.
4- Edge – this connects one edge to another if all conditions are satisfied.

The Root and Internal Node contain the attribute test conditions. The attribute test will test the conditions which will check the characteristics of each record and separates them.

2) CART

The CART is a type of decision tree classifier that uses the Gini index for its splitting feature [10] it divides the feature space into sets of rectangular parts that are disjointed.

3) Gini Impurity

Gini impurity describes the Node's sample composition degree of purity. The formula that describes the Gini impurity is:

$$G = 1 - \sum_{K=1}^{n} P_k^2$$

P_k : refer to instances in the Node; the ratio of class *k* [10].
The classification and CART algorithm work together to separate one Node into two Nodes as the parent's children. Minimising impurity cost function is used to select splitting criterion that are consisting attributes such as threshold of the feature and the feature itself.

$$J(k, t_k) = G_{left} \times m_{left}/m + G_{right} \times m_{right}/m$$

The weighted sum will make the algorithm stop when the Tree has reached its maximum depth or if the weighted sum is bigger than its parent's Gini impurity during the split.

B. K-nearest Neighbours Classifier

K-nearest Neighbours' Classifier uses a non-parametric method for any form of classification. In the feature-space the input will have several closest training examples (k). Therefore, it uses the distance metric to select. The class of the data will be determined by weighted voting where the neighbours are requested to vote on each class. "The votes are weighted by the inverse of their distance to the query" [11]. The formula used for this voting method is:

$$Vote(a_i) = \sum_{c=1}^{k} \frac{1}{d(q, x_c)^n} * 1(a_j, a_c)$$

The classifier is described by [11] to have a transparent process which allows the implementation to be easier.
Hence, it has compatibility with noise reduction techniques where they are effective in improving its accuracy. Whereas, the disadvantages of this method include run-time performance issues if the training data is large . In comparison to Random Forests classifiers, it will underperform. Therefore, a precise feature selection is advised.

C. Support Vector Machine Classifier

1) Support vectors

The data-points that are coordinated nearest to the decision surface or the hyperplane is called support vectors [12]. The data-points the most difficult to classify on the optimum region of the decision surface; the support vectors have direct bearing.

2) Support Vector Machine

Data-points are separated by a hyperplane, the SVM maximises the margin around it as shown in figure 1.

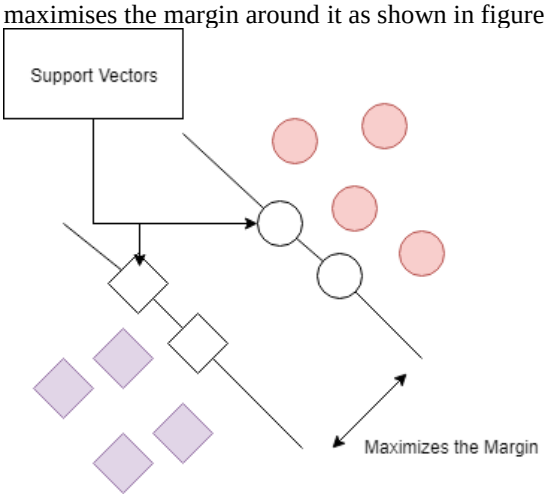

Figure 1 SVM in process

The margin in statistical learning has an important role.
For example, if we have a hyperplane:

$$< w^*, x > + b^* = 0, ||w^*|| = 1$$

That hyperplane is called γ-margin separating hyperplane only if:

$$y_i(< w^*, x > + b^*) \geq \gamma$$

Where γ is the size of the margin.
Therefore, for any data point (i) with coordination of (x_i, y_i), the formula above must apply for any separating hyperplanes with γ>0, the hyperplane can be converted into this form [12].

In the maximising process the total distance of two hyperplanes is $\frac{2}{||w||}$. To maximise the margin, it is required for the ||w|| to be minimised with only one condition where there are no data-points in between two hyperplanes.

D. Random Forests Classifier

1) RF overview

Random forest is a supervised learning algorithm that uses an ensemble of classification trees for classification. The algorithm is advised to be used with bagging methods by [2].

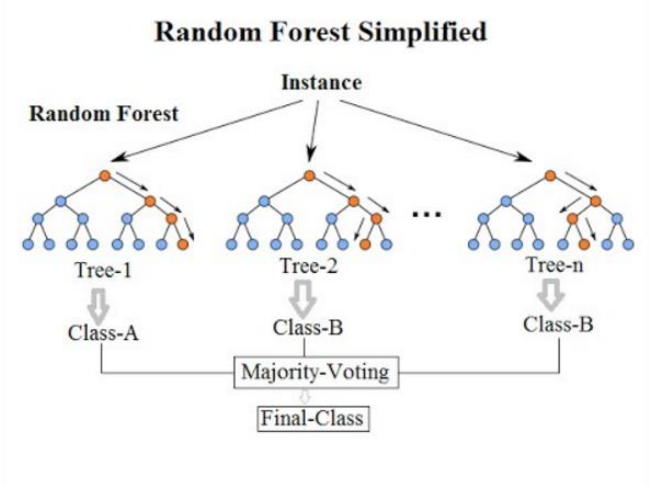

Figure 2 Random forest tree structure Taken from [8]

In the process of the Random Forests, the algorithm will choose randomly from the subset of the training set and will then create a set of decision trees from the randomly selected set. The votes from different decision trees are aggregated and then the candidate vote will decide the final class of the data. Furthermore, the Random Forests algorithm uses random feature selection which allows each tree to iteratively grow. This process results in having low variance and bias. The bagging and bootstrap aggregation will also allow the tree to have minimum correlation and have an incremental manner in building the classifiers. Stacking method in ensemble models will have a role in training the model by combining prediction of two or more preceding models. One of the differences between decision trees and Random Forests is that Random Forests prevents decision tree's common error which is "the overfitting". This error occurs when the classifier models train the data too well or when the model trains the limited set of data to an extent where the model will limit itself from improving. Random Forests accept and run efficiently on large datasets. The size of the dataset will not affect the accuracy of the classifier and can achieve higher accuracy in comparison to other classifiers. As mentioned in the introduction, the ensemble method or Random Forests is quite computationally expensive to use. The Random Forests will logically have multiple hidden layers as being a collection of decision trees made by an algorithm. The number of layers or hidden layers will affect the evaluation time of the model, therefore the higher the number of layers will affect the time for the process to be done. The increase of the O_n or the time complexity to evaluate will also depend on

the number of decision trees in the Random Forests. As it was explained by [1], the complexity and the large size of the model can be interpreted as the strength of the model. However, [1] further explained the model obtained is quite cumbersome and computationally expensive which results in difficulties when using the model.

E. Breiman's proposition

Conclusively, the Random Forests method is the main method representing bagging [2].
[2] explains Random Forests as a classifier that consists of a set of tree-structured classifiers that Represents the formula:

$$\{t(x, \Theta b), b = 1, \ldots, B\}$$

The $\{\Theta b\}$ are independent identically distributed random vectors [2].
Each tree built will return a vote value for the candidate class at input *x*. [2] proposed two different methods as a solution to the classification problem.
The first suggestion was to use bagging, random feature selection and random split decision [2] .
The method suggested by [2] made the features of training set to have randomness in the structure. Moreover, a voting that was unweighted to assign class to features was advised in the [2] propose.
Comparatively, the *Adaboost* by [5] and *Margin Classifiers* by [6] was mentioned by [2] to use an algorithm to iteratively reweight the training set of data with exclusion of any randomness in the process. Furthermore, the algorithm suggests using weighted voting to predict the test set of data's classes.

F. Voting methods used in ensambling

[3] explained, Candidate vote can also be described as majority voting. In majority voting, each classifier will be called to predict the class of an unlabelled data. Whilst the votes are gathered, the class with the highest number of votes will be returned as the class of the unlabelled data. Comparatively, there is another method to choose the class, this method goes by Veto voting which is explained by [4], in this concept each classifier can veto other classifier's decisions.

G. Knowledge distillation

1) Overview

Recently [7] introduced the concept of knowledge distillation. The concept was quite effective in training set of data that had small network models. The model affected the time and computational power. The concept has enabled classification to be processed sufficiently fast and required less memory. The concept mentioned is the knowledge distillation method. Knowledge distillation allows small network (student) to learn from a larger network (teacher) and creates a deep network [7]. In the training, it uses the "teacher-student" method where a large pre-trained model (teacher) is imitated by a small model (student). The knowledge is set to be transferred from the teacher model to the student model. This action is done by minimising a loss function. The distribution of class probabilities achieved from the teacher model described as the target for the training.
A distribution probability of the classes is achieved from the teacher model. For example, in case of classes such as: [class_1 , class_2, class_3,class_4], the teacher model will create a list of probabilities of the classes in form of:
[p_1, p_2, p_3, p_4].
In most cases this distribution is not quite efficient as it seems. For example, in some cases the distribution suggests one class has very high probability and others have probabilities close to zero for instance:
[0%,1%,0.5%,70%].
Therefore, no additional information gained from this action beyond the information (ground truth) where labels already provided in the dataset.
"*Softmax temperature*" was introduced by [7]. [7] suggests that the probability of the class is calculated from the logits as:

$$p_i = exp(z_i/t) / \sum_j exp(z_j/t)$$

The t in the formula stands for the temperature. t will have no influence in the result and it will return the standard *softmax* function if t is set to 1 in the formula.
The probability distribution generated by the *softmax* function will become softer when temperature is increased.
Therefore, this action will provide more information beyond the "ground truth" provided by the preceding dataset.

2) Distillation loss

The "*distillation loss*" is a loss occurred in the computation of the loss function and the teacher's soft targets. In the computation, the same temperature value and formula is used to compute the student's logits.
The overall loss functions will have distillation and student losses playing an important role in the formula below:

$$L(a; W) = \alpha * H(b, \sigma(z_s; T = 1)) + \beta * H(\sigma(z_t; T = \tau), \sigma(z_s, T = \tau))$$

Where:

- a is the input.
- W indicates the student model parameters.
- b indicates the ground truth label.
- H is the cross-entropy loss function.
- σ is the softmax function
- T is the temperature which parameterises the softmax function
- α *and* β are the coefficients.
- z_s are the student logits.
- z_t are the teacher logits.

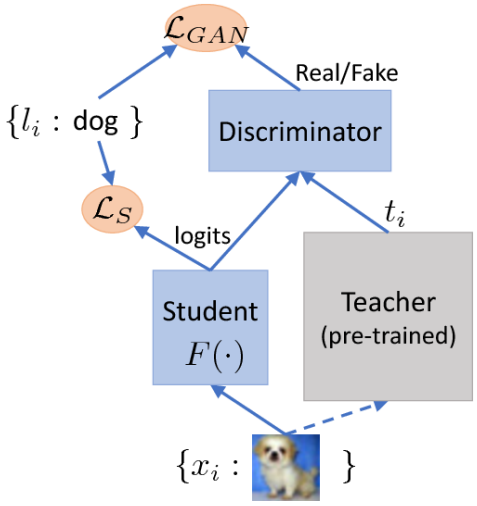

Figure 3 knowledge distillation architecture. Taken from [9]

III. Methodology

A. Data prepration

The datasets have changed since the project proposal. Therefore, new datasets are taken from UCB online [13]:

1- Car dataset [14]
2- Bank dataset [15]
3- Iris dataset [16]

The datasets are prepared and stored in a large data structure, this structure holds all the datasets provided, for instance [dataset_1, dataset_2, dataset_3,ect...] is a good representation of what is stored in the array of datasets.
The labels in each dataset will be encoded using *LabelEncoder* from *sklearn*. The reason for encoding the labels is due to the model chosen only works with numbers.
Unfortunately, it is not quite simple. Because the encoded data must be decoded back for many analytic purposes such as legends, confusion matrices and classification reports. Therefore, a set of useful encoders will be collected from the encoder list as shown in Figure 4.

```
encoders = [] # List of usefull encoders
for j, dataset in enumerate(datasets):
    length = len(dataset.columns)
    for i in range(length):
        if pd.api.types.is_string_dtype(dataset.iloc[:, i]):
            le = LabelEncoder()
            dataset.iloc[:, i] = le.fit_transform(dataset.iloc[:, i])
            if i == length - 1 or dataset.columns[i] == 'safety':
                encoders.append(le)
```

Figure 4 encoding code snippet

The encoded data will first be transformed into a collection of x and y for each dataset and then each dataset will be split into train and test dataset as shown in Figure 5.

```
data = ( # Collection of x and y for each dataset
    [datasets[0].iloc[:, :-1], datasets[0].iloc[:, -1]],
    [datasets[1].iloc[:, :-1], datasets[1].iloc[:, -1]],
    [datasets[2].iloc[:, :-1], datasets[2].iloc[:, -1]])

# Split each dataset into train and test datasets
datasets[0] = list(train_test_split(*data[0]))
datasets[1] = list(train_test_split(*data[1]))
datasets[2] = list(train_test_split(*data[2]))
```

Figure 5 Train and test division code snippet

B. RF classifier in training the data

A random Forest classifier will be used to train each dataset with a reasonably high number of trees (100).

C. Customised binning method

A *numpy.histogram* binning method was suggested to use to store the probabilities. This method was not quite right for the model chosen, therefore, a customised binning method was created.
The probabilities were first gathered from the trained RF. The data structure will have a shape like: [p_1,p_2,p_3,etc..]. therefore, with an iteration each probability can be used to store in a variable called "bin" then a set of conditions will be applied. These conditions are:

```
if bin != 1.0:
    if bin == 0.1 * int(10 * bin):
            bin += 0.1
    bin = round(0.1 * ceil(10 * bin), 1
)
```

After this process is done with all the probabilities, a new form of target data is created. LabelEncoder is again used to encode the newly created target data.

D. Train decision tree on the distillited data

A decision tree classifier chosen to train on the labels from the original dataset but replace the targets with the new targets.

E. Train decision tree on the original data

To experiment with distilled data another classifier is used to train on the original dataset excluding any distillation. This will show if there are any improvements or changes and will be compared further in the Evaluation part of this report.

F. State-of-art methods used in training the original data

State-of-art methods such as K-nearest neighbours and SVM are used to train the original datasets as the accuracy will be compared with the other methods explained.

IV. Experiments

Each classifier mentioned in the Methodology section will use plot_boundry function defined in the project to visualise the performance. Moreover, the decision tree classifier plot_decision_tree function was written to visualise the best route.
To plot decision boundaries of each decision tree classifier trained on distilled data. Number of features were limited to two for each dataset to plot as a 2D plot where x and y axes would be two of the most important features and each class would have its own colour.
To achieve this, the RFE (Recursive Feature Elimination) function from the sklearn library was used.

1) Car dataset experiment

	Accuracy on Original Dataset (%)
Random_forest	97.69
Decision_Tree	98.61
Distilled trained Decision_Tree	98.61
KNC	91.9

SVC	82.87

Table 1 Accuracy table on car dataset

Visualisation of the decision tree trained on the distilled Car dataset is shown on Figure 6.

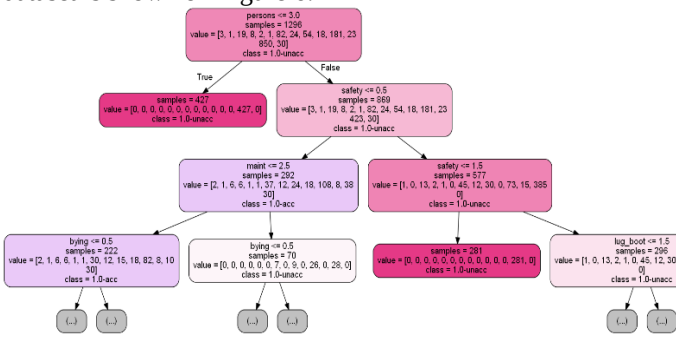

Figure 6 decision tree on Car dataset

Visualisation of the decision boundaries of the decision tree classifier trained on the distilled car dataset are shown in Figure 7.

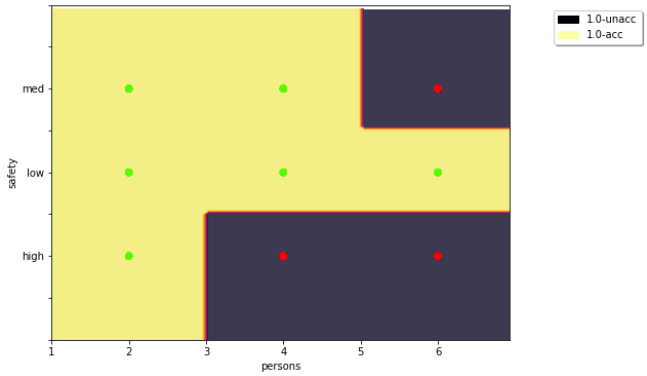

Figure 7 decision boundaries of DT trained on the distilled car dataset

2) Bank dataset experiment

	Accuracy on Original Dataset (%)
Random_forest	89.12
Decision_Tree	85.94
Distilled trained Decision_Tree	85.59
KNC	87.62
SVC	88.86

Table 2 Accuracy table on Bank dataset

Visualisation of the decision tree trained on the distilled Bank dataset is shown on Figure 8.

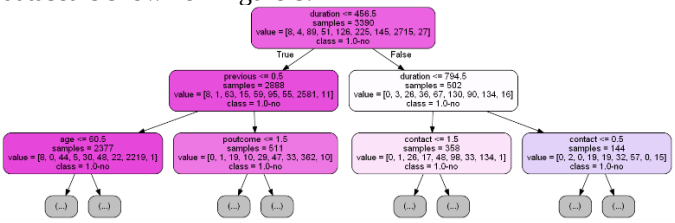

Figure 8 DT on distilled bank dataset

Visualisation of the decision boundaries of the decision tree classifier trained on the distilled Bank dataset are shown in Figure 9.

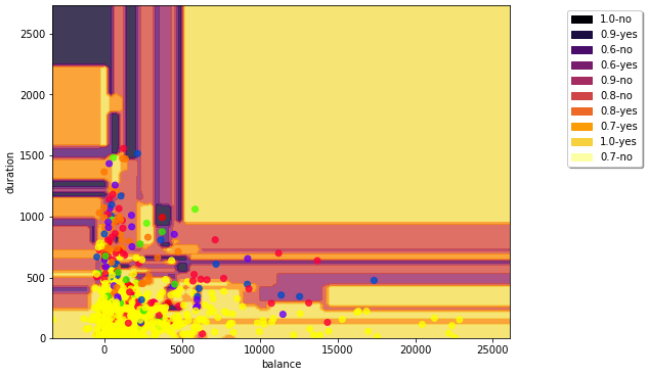

Figure 9 decision boundaries of DT trained on the distilled Bank dataset

3) Iris dataset experiment

	Accuracy on Original Dataset (%)
Random_forest	92.11
Decision_Tree	94.74
Distilled trained Decision_Tree	94.74
KNC	94.74
SVC	97.37

Table 3 Accuracy table on Iris dataset

Visualisation of the decision tree trained on the distilled Bank dataset is shown on Figure 10.

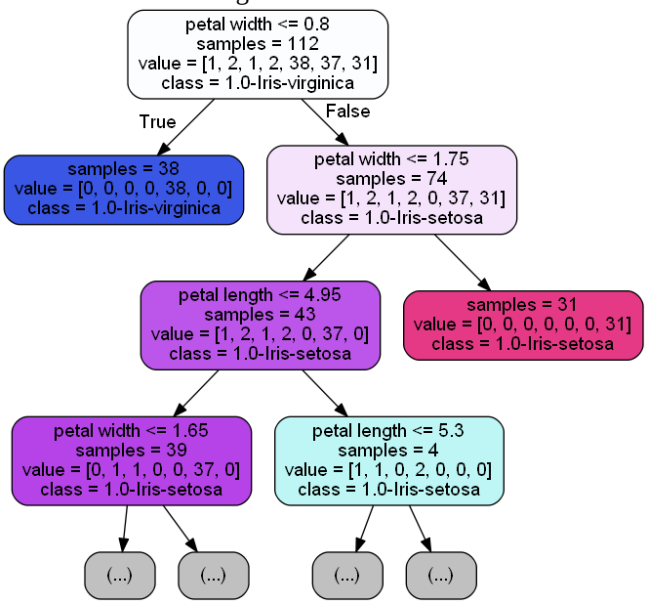

Figure 10 DT on distilled Iris dataset

Visualisation of the decision boundaries of the decision tree classifier trained on the distilled Iris dataset are shown in Figure 11.

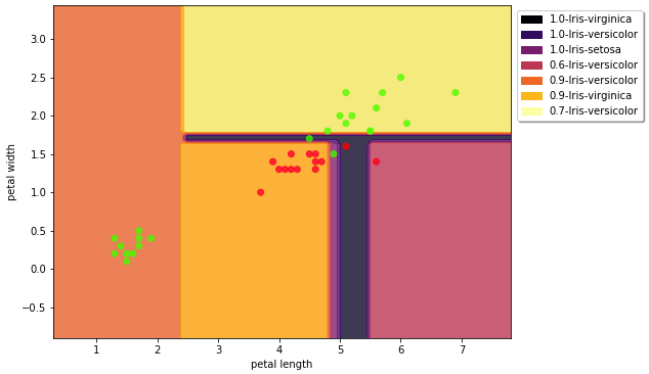

Figure 11 decision boundaries of DT trained on the distilled Iris dataset

V. Evaluation

After we achieved all the probabilities, we used the knowledge distillation observed in the background section to create a New Dataset to reduce the size and complexity of the Dataset. A Decision Tree Classifier was chosen to train on the New Dataset. The aim is to create a Dataset that is more usable than the previous one, the accuracies must be close to the Decision Tree training on the Original Dataset. Regarding the Car and Iris dataset, there were no differences observed between accuracies achieved by the Decision Tree Classifier. Training on the classifier for the Original Dataset and the Distilled Dataset achieved 98.61 percent for the Car Dataset and 94.74 percent for the Iris Dataset, whereas on the Bank Dataset the Decision Tree which trained on the Distilled Dataset achieved lower accuracy than the Original Dataset by 0.35 percent. The Bank Dataset has 10 separate targets, while the Car Dataset has 2 separate targets and the Iris Dataset contains 7 separate targets. A correlation between the accuracy of the Classifier is observed with the number of targets they have after the Distillation process; as observed it has performed uniquely the same on the Car and the Iris Datasets. However, the higher number of targets will challenge the classifier to achieve a higher level of accuracy.

Moving on to the K-neighbours (KNC) and SVM classifiers accuracy on the Car Dataset. KNC achieved 91.1 percent which is lower than the decision tree trained. Therefore, a further investigation was taken to see the classification report of KNC.

~~~~~~~~~~~~~~ Classification report ~~~~~~~~~~~~

	precision	recall	f1-score	support
vgood	0.83	0.85	0.84	99
acc	0.57	0.29	0.38	14
unacc	0.95	0.99	0.97	304
good	1.00	0.60	0.75	15
micro avg	0.92	0.92	0.92	432
macro avg	0.84	0.68	0.74	432
weighted avg	0.91	0.92	0.91	432

Figure 12 classification report on KNC

The f1-score takes both false negative and false positive into account therefore it shows a quite high achievement in most of the classes beside the accessible class which achieved 0.38.

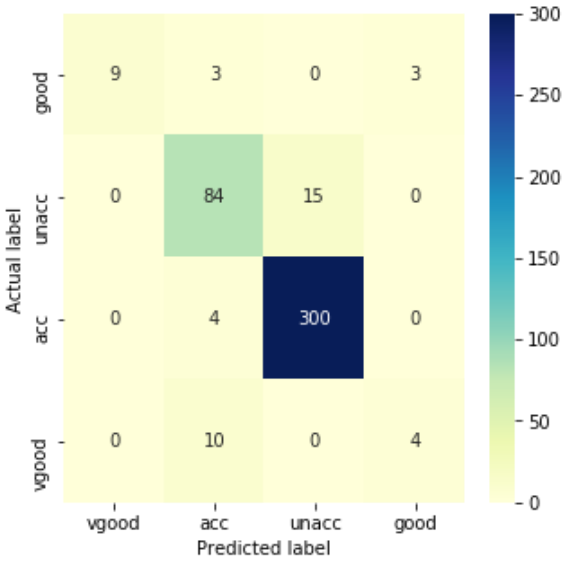

Figure 13 confusion matrix using KNC on car dataset

The SVC achieved 82.87 percent on the Car dataset, this is lower than KNC and decision tree trained.

~~~~~~~~~~~~~~ Classification report ~~~~~~~~~~~~

	precision	recall	f1-score	support
vgood	0.77	0.48	0.60	99
acc	0.00	0.00	0.00	14
unacc	0.83	1.00	0.91	304
good	1.00	0.47	0.64	15
micro avg	0.83	0.83	0.83	432
macro avg	0.65	0.49	0.54	432
weighted avg	0.80	0.83	0.80	432

Figure 14 SVC on bank dataset

As shown in Figure 14, f1-score for accessibility was 0.0. In comparison to other datasets; Figure 16 show that SVC achieved 0.0 on *no* classification in f1-score. And KNC had all low scores in *no* class in f1-score.

~~~~~~~~~~~~~~ Classification report ~~~~~~~~~~~~

	precision	recall	f1-score	support
yes	0.90	0.96	0.93	1005
no	0.38	0.18	0.25	126
micro avg	0.88	0.88	0.88	1131
macro avg	0.64	0.57	0.59	1131
weighted avg	0.85	0.88	0.86	1131

Figure 15 KNC score on Bank dataset

~~~~~~~~~~~~~~ Classification report ~~~~~~~~~~~~

	precision	recall	f1-score	support
yes	0.89	1.00	0.94	1005
no	0.00	0.00	0.00	126
micro avg	0.89	0.89	0.89	1131
macro avg	0.44	0.50	0.47	1131
weighted avg	0.79	0.89	0.84	1131

Figure 16 SVC score on Bank dataset

Whereas, the Iris dataset had a very high f1-score for both SVC and KNC as shown in Figure 17 and 18.

~~~~~~~~~~~~~ Classification report ~~~~~~~~~~

	precision	recall	f1-score	support
Iris-virginica	1.00	1.00	1.00	12
Iris-setosa	1.00	0.82	0.90	11
Iris-versicolor	0.88	1.00	0.94	15
micro avg	0.95	0.95	0.95	38
macro avg	0.96	0.94	0.95	38
weighted avg	0.95	0.95	0.95	38

Figure 17 KNC score on the Iris dataset

~~~~~~~~~~~~~ Classification report ~~~~~~~~~~

	precision	recall	f1-score	support
Iris-virginica	1.00	1.00	1.00	12
Iris-setosa	1.00	0.91	0.95	11
Iris-versicolor	0.94	1.00	0.97	15
micro avg	0.97	0.97	0.97	38
macro avg	0.98	0.97	0.97	38
weighted avg	0.98	0.97	0.97	38

Figure 18 SVC score on Iris dataset

All the classifiers are compared in Figure 19, the graph suggests that when excluding the Random Forests from the table the SVC has the highest accuracy in the Bank and Iris datasets. Whereas, the Distilled trained DT and normal decision tree had the highest score in classifying the Car Dataset. This prove that SVC has done worse than the other classifiers on the Car dataset.

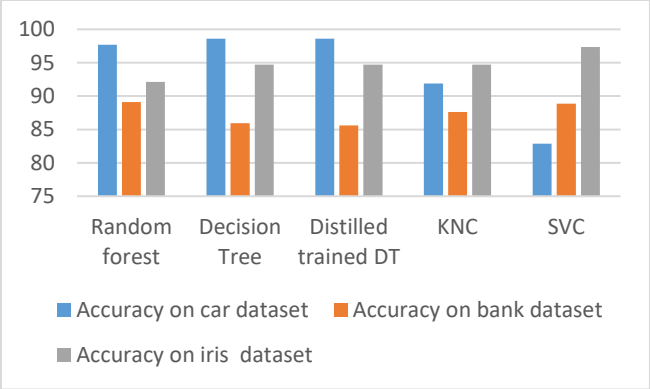

Figure 19, Accuracies compared between different datasets

VI. Conclusion

From all the observations obtained from the data, it was found that the accuracy of the classifier is directly under influence of the dataset. The dataset with the higher number of targets will increase the number of Nodes for decision trees to choose from. This concludes that a stricter decision tree with a higher number of conditions should be chosen to improve its accuracy. The Distilled Dataset improved the data structure to make it usable, as well as with the encodings it made it possible for the labels to convert into a form that could be used in the classifiers. The overall accuracy of the decision tree and distilled decision tree was kept the same during the experiment which allows the O_n to be less than the decision tree classifying the original data. Comparatively, the KNC had a lot better accuracy on average between the different datasets chosen. However, the limited time for this assignment forced me to choose other datasets for comparing the distillation results instead of the three assigned. Moreover, the python language does not have any algorithms to calculate the time it takes for the classifiers to assign class to labels. The O_n or the time complexity was not added to the report due to limitation of the libraries used.

References

[1] G. Papamakarios , “Distilling Model Knowledge”, 2015.

[2] L. Breiman,“Bagging predictors Machine Learning”, vol. 26, pp. 123-140, 1996.

[3] L. Lam and S. Y. Suen, "Application of majority voting to pattern recognition: an analysis of its behavior and performance," in *IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans*, vol. 27, no. 5, pp. 553-568, Sept. 1997. doi: 10.1109/3468.618255

[4] Shahzad, R. K., & Lavesson, N. 2012. “Veto-based malware detection”. In 2012 seventh international conference on availability, reliability and security (ARES), Prague, Czech Republic (pp. 47–54). New York City, NY: IEEE.

[5] Y. Freund, R Schapire, "Experiments with a new boosting algorithms Machine Learning", Proceedings of the Thirteenth International conference, pp. 148-156, 1996.

[6] Mason, Llew, Peter L. Bartlett, and Jonathan Baxter. "Improved generalization through explicit optimization of margins." Machine Learning 38.3 (2000): 243-255.

[7] G. Hinton, O. Vinyals, J. Dean ,”Distilling the knowledge in a Neural Network”,2015.

[8] "Random Forest Simple Explanation", *Medium*, 2019. [Online]. Available: https://medium.com/@williamkoehrsen/random-forest-simple-explanation-377895a60d2d. [Accessed: 24- Apr- 2019].

[9] Z. Xu, Y. Hsu and J. Huang, "Training Shallow and Thin Networks for Acceleration via Knowledge Distillation with Conditional Adversarial Networks", *Semanticscholar.org*, 2019. [Online]. Available: https://www.semanticscholar.org/paper/Training-Shallow-and-Thin-Networks-for-Acceleration-Xu-Hsu/86448ec1e48740163567e494592bb699d4a1900f/figure/4. [Accessed: 24- Apr- 2019].

[10] P. Tan, M. Steinbach, V. Kumar and A. Karpatne, *Introduction to Data Mining, Global Edition*. Harlow, United Kingdom: Pearson Education Limited, 2019.

[11] Cunningham, Padraig & Delany, Sarah. (2007). k-Nearest neighbour classifiers. Mult Classif Syst.

[12] Fradkin, Dmitriy & Muchnik, Ilya. (2006). Support vector machines for classification. DIMACS Series in Discrete Mathematics and Theoretical Computer Science.

[13] "UCI Machine Learning Repository", *Archive.ics.uci.edu*, 2019. [Online]. Available:

https://archive.ics.uci.edu/ml/index.php. [Accessed: 24- Apr- 2019].
[14] "UCI Machine Learning Repository: Car Evaluation Data Set", Archive.ics.uci.edu, 2019. [Online]. Available: https://archive.ics.uci.edu/ml/datasets/Car+Evaluation. [Accessed: 24- Apr- 2019].
[15]"UCI Machine Learning Repository: Bank Marketing Data Set", Archive.ics.uci.edu, 2019. [Online]. Available: https://archive.ics.uci.edu/ml/datasets/Bank+Marketing. [Accessed: 24- Apr- 2019].
[16]"UCI Machine Learning Repository: Iris Data Set", Archive.ics.uci.edu, 2019. [Online]. Available: https://archive.ics.uci.edu/ml/datasets/Iris. [Accessed: 24- Apr- 2019].