Hybrid REST based Machine Learning MicroService for Campaign Advertisers

Ashay V Sant, Shraddha Khonde
Modern Education Society's College of Engineering Pune, India
Modern Education Society's College of Engineering Pune, India
(E-mail: ashayvsant@gmail.com, shraddha.khonde@mescoepune.org)

***Abstract*—**Designing a generic machine learning model by using hybrid approach using scikit-learn and Google's Tensor ow in python. The aim of the model is to predict the Global Advertiser associated with a particular client-code given an advertising agency and other feature vectors like Client name, unique Id. In the advertisement world there are 3 entities namely the advertisement agencies, the sellers like Google and Yahoo. Advertisers include Coke, Nike. The advertisement agencies create the campaigns on behalf of advertisers and place the media buys on the seller side. This integration needs machine learning to enhance user experience. The hybrid microservice is a Django rest based python service. This will enable the communication via rest API. The models could be stored in shared cache. Hazelcast provides a python client as well to store the data into cache as a HashMap. There can be multiple paths through the service and multiple data types. The textual data is converted to tokens and then to a tensor matrix using tensor ow tokenizers. Categorical data works well with scikit-learn but a customized preprocessor is used for Tensorflow.

Keywords*—Machine Learning, Google Tensorflow, scikit-learn, python, microservice, hazelcast, hazelcast-python-client, delegator, abstraction, design patterns, campaigns, Sellers, Advertisers, Advertiser Agencies, REST (representational state transfer), Ensemble of classifiers, Categorical Data, prediction*

I. INTRODUCTION

The dataset that is generated on production environments for any application could be used for machine learning and enhancing the user experience for any platform. For this behavior we need a generic Machine Learning service which can be used to communicate with various other microservices. There is a structured way to do this using Django and python. Django has got similar properties analogous to Spring Framework in Java. Rest protocol based communication could be facilated using Django framework in python. This standardizes and makes the service to be deployed easily. The Django enabled service could also be deployed onto kubernetes containers. There are other alternatives to tensor ow and scikit-learn. The whole idea is to all together use a different framework to the likes of Amazon Sagemaker, KNIME studio or Microsoft Azure ML Studio. These all the frameworks are not open source although leverage the amount of time to get the model deployed.

Bayes classifier is popular algorithm used for text classification. In this paper, the authors have predicted the song performer based on lyrics only. The precision which was achieved was 93% and Recall of 95%. Also F1 measure of 94% was achieved.

The problems stated include creation of the dataset, guest appearances of music performers needed to be handled with care [1].

The authors have described multiple machine learning algorithms as part of this paper. The authors focus on predicting the future using machine learning. Here by future we mean unknown data-set. The algorithms described by author include

1) *Decision-Tree*
2) *Support Vector Machine*
3) *K Nearest Neighbor Classifier Naive Bayes Algorithm*
4) *Linear Regression*

The author also describes the advantages and disadvantages of each and every algorithm in detail. Author states that Linear Regression could not be applied to non-linear dataset, however it is much easy to understand and design. Bayes algorithm is simple, fast and scalable. It is best suitable for text classification. It assumes the concept of class conditional independence [2].

A multiTree algorithm is generated for Intrusion detection system. Alongside multiple algorithms are used like that of Decision Tree, KNN, DNN, Random forests etc. to learn an ensemble classifier and perform majority voting [3]. Steps involved in the algorithm include

The training data-set is fed

Standardization of data by preprocessing module Ensemble training of classifiers

Use of cross-validation for training all classifiers. Also boosting is performed for increased accuracy. The author makes a mention for the TrResampling whereby a new training data-set is created from the existing dataset. The size of this new data-set is same as that of original data-set. The data in original data-set is weighted and selection is based on these weights. The weights are arranged such that more frequently the data appears in the new pseudo training-data set, the less likely it is to be misclassified. TrAdaBoost strategy is used to influence the inclusion of the training-data-set into the pseudo

training data-set. The author also used Bagging and Boosting algorithms in conjunction with TrResampling. These relate to the ensemble model [4].

Meaning of High availability on oracle clusters is explained by the authors and the paper is from Oracle Corporation USA directly. According to the authors high availability means that when one instance on a node is down, due to some hardware failure or some software failure then there is process named instance recovery that runs. The failure node that has actually failed due to some fault, are selected and undergo a process called recovery. But the request if reaches such to be repaired node, then we have the problem of failure of the request hence all requests to the node that is potential candidate of being repaired should be blocked. The paper explains the buddy system for the system recovery for the failed node. The buddy service introduced in 12.2.0.1 Oracle release to avoid the problem of resource identification of resources that need to undergo the process of recovery. The buddy system says that each node in the cluster is assigned another node that acts as buddy for the said node. There is a one-one mapping between a buddy and a particular node. The buddy continuously scans the node for potential redo log using the buddy instance num read buffers parameter. Thus the buddy will have the entire redo log with itself for a particular node. Consider a scenario that when the node fails, its buddy has an entire redo log scanned with it. Hence the recovery can be made quickly as the buddy has the recovery set scanned for the failure node. Consider that the buddy has not fully scanned the log, before recovery it scans the log completely before doing a recovery as incomplete log may have an incomplete instance coming up. If the recovery candidate is not the buddy then the entire scan of the redo log needs to be done. Then the said recovery instance, having with it, the redo logs of multiple failed instances, has to combine all the redo log using union set operation. Thus this paper gives us a basic insight into the high availability speed up using the buddy system which keeps track of the redo logs regularly [5].

This white paper aims to tell us in detail about the timestamps and time series data that comes from real-time databases. Consider a sensor reading the temperatures of a 2-wheeler engine. Thereby the sensor records the timestamp as an additional parameter along with the said temperature. This may increase the numerosity of the data that is collected. All this data is not necessary for the warehousing and data mining tasks. Only the data within the particular is necessary for the further computations. To accomplish this the oracle family gives us with Time to live feature. The oracle database has a feature which allows the data to live in the database for a particular amount of time only. After that the data is automatically removed and new data takes its place. row.setTTL (TimeToLive. noofDays (5)); this sets the time to live for a particular row to 5 days. That means that after 5 days the data will be erased from the data store. Note this feature is strictly being talked about related to Oracle Databases and specifically Oracle 12c No SQL. Advantages illustrated in the paper regarding the Time to Live Feature Reduces the size of the database. When dealing with large amounts of such temporal data, the underlying database should also look at the performance aspects. Thus we have the Oracle NoSQL's smart client driver. This smart client driver always routs the data query to the most efficient node. This ensures low latency. Also we have secondary indexing in oracle databases that is on secondary key column. These indexes reside on the shard and are local to a particular shard. These shards can be searched in parallel once the query is generated for data thus increasing the response time of the query [6].

The literature tells us about undo and redo operations for database recovery in high availability environment. Checkpoints are like restore points in Microsoft windows wherein the systems state is stored just like a snapshot and it is restored as and when needed. The paper suggests an approach of Roll Forward like that of Roll back. We have to save the entire state of the VM in traditional methods. This costly transaction is addressed by the authors which gets rid of pausing the VMs for saving their entire state. The authors stress on storing only the next instruction to be executed and all the data structures that help in restoring any failed transaction [7].

The authors have described highlight the below techniques

1) *Data Training and Testing on same data*
2) *Dataset separation into training and testing parts Cross Validation*

In k fold cross validation the test is carried out k times. If k is set to 7 then evaluation process will be done 7 times using the different data each time. Also the sampled data is such that at the end of k runs entire dataset would have been evaluated. The machine learning algorithms investigated in this literature involve

1) *Logistic Regression k nearest Neighbor Decision Tree*
2) *Naive Bayes*
3) *The datasets tested by the authors include*
4) *Fire Dataset (Air quality, temperatures for classification)*

Blood Transfusion Service Center Dataset (Time since donation, Donation Frequency, total blood donated)

Iris Dataset (Iris flower data set. Iris species of plants are investigated)

Lenses Data Set (hard contact lenses, soft contact lenses, no contact lenses) Evaluation Metrics used include

1.0.1. Accuracy

$$\frac{TP + TN}{TP + FP + FN + TN}$$

1.0.2. Precision

$$\frac{TP}{TP + FP}$$

1.0.3. Recall

$$\frac{TP}{TP + FN}$$

The author concludes that in each of the data-set different models perform differently and these is no one model that could be used for each and every dataset. This boosts our decision for using ensemble of classifiers so that we could leverage the bene t from all algorithms [8].

This paper describes optimization techniques to reduce the training time. Also shallow learning networks that do not have as many layers as deep Neural networks are not suitable for the large and multidimensional data sets that we have nowadays. As an example consider the task of recognizing the image using CNN (Convolution Neural Network). The low level pixel details from an image are taken and analyzed. Then mid-level features including the Shapes and edges along with patterns observed and further analyzed. DNN is implemented using CNN.

CNN is primarily used in Image Processing. CNN takes input of 2D image and multiple layers of Neural Networks call filters or kernels are present. The mapping is to only specific neurons included in the spatial locality. This reduces chances of overfitting. The final layers in the CNN are fully connected and thus are responsible for classification [9].

The ultimate aim in Machine Learning is to learn a model from the dataset. This data-set is called as training data-set which is stored in relational databases. This can come from highly available database as well. When we are considering the application of highly available databases, Machine Learning is a strong source. The dataset is stored in relational databases and Hyper-Box approach tries searching the database and analyzing it using queries. Consider the example suggested by the authors of detecting k-nearest neighbors. In this case we need to identify the k nearest neighbors. This takes a scan through the data after it has been retrieved. Why not accelerate it at the time of querying itself? If we put the distance metrics in the where clause of the SQL Query we could just filter out the neighbors at the relational database level, thereby accelerating the Machine Learning process using KNN [10].

Database queries are quite complex for new users. Hence using a natural language processing approach these can be made simpler. Consider the query spelled as a sentence. After this there are following steps involved

1) *Token Analyzing Spelling Correction*
2) *Ambiguity elimination Tagger*
3) *Morpheme*

Syntax analysis and Context Free Grammar analysis using Tree Translation to XML

Translation to Database Query

This way the end-user just needs to type a sentence and then automatically the query processing engine will translate the sentence to the query using intermediator XML syntax. This way we could use relational highly available database and highly available microservices to translate Natural Language to Query as requested by the end-user [11].

A. *Leading Solutions*

1) *KNIME*

KNIME has knime-server on AWS and Microsoft Azure platforms and allows to deploy directly a rest based model. KNIME Analytic Platform is open source but the knime-server is not. To deploy the model, there is licensing cost.

2) *Microsoft Azure*

It has got 1 month free trial and post that there are free services that free all the way. However there is limitation of 1 hour per experiment and production cost is as per the number of times the API is hit

3) *Amazon Sagemaker*

This is powered by amazon and allows to create, evaluate and deploy Machine Learning models. Sagemaker provides free trial of 2 months and then charges as per the usage.

B. *The Open Source Alternatives*

1) *Tensorflow*

Allows training of deep neural network in python and JavaScript. It is an end-to-end machine learning platform. It allows training of deep neural network models easily.

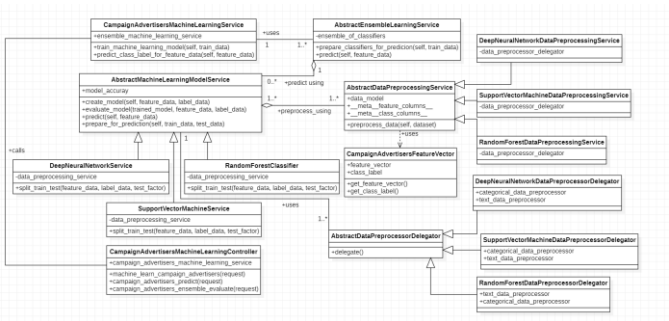
2) *Scikit-learn*

It is a free machine learning platform for python. It behaves very well and is production friendly. It also provides automated handling for categorical data. It should scale well with the tensor ow tokenizers as well.

II. DESIGN AND ARCHITECTURE

The primary architecture is microservice based wherein the communication is stateless rest protocol (Might be staeful in future if authentication is coupled between the client and the server side). Django framework is used for handling Rest requests and also provides an Object Relation Mapping framework.

FIGURE I. CLASS DIAGRAM



A. *Rest Calls*

Allows stateless communication between the client and the server. Rest call can be made to service from client side using

JavaScript or using application like postman when testing. There are 2 REST calls for machine-learning-service.

1) *GET call to get the training data*
2) *PUT call for prediction*

A controller is exposed to listen to the rest requests via the Django framework. The URL and the method are connected together in the Django URL patterns

B. Feature Vector Definition

The feature vectors can be defined in a predefined structure in a JSON schema. Herein the feature vectors can be grouped together into a dictionary. This dictionary can again map to the actual database entity. This enables to dynamically add or remove a feature vector column. Also the data type of the feature vector columns can be changed. However the service needs to be redeployed to make this change visible. Currently two types of data are supported

1) *text*
2) *categorical*

C. Data Types of Feature Vector

There is separate preprocessor for each model as it is heterogeneous service consisting of both scikit-learn and google tensor ow. All the preprocessors inherit the Abstract Preprocessor. Individual processor has to implement all the abstract methods and in turn may call the super method directly if the preprocessing is not needed. Text preprocessors handle tokenization whereas the categorical pre preprocessors handle the categorical data. In case of tensor ow a customized logic is put into place. The scikit-learn does not need any pre-processing and handles categorical data as is in the form of strings

D. Data Preprocessing

There is a PreProcessingService associated with every MachineLearningModelService. This pre-processing service has the responsibility of getting the feature vectors from custom JSON File and delegate the pre-processing part to the appropriate CategoricalDataPreprocessor or TextDataPreprocessor. Each learned model has to have a different Delegator just for separation of concerns and uniformity in the structure.

E. Model creation and storage

Each model is created and stored in an object. This model is persisted per instance of the service. When the service is restarted the model will be lost. To store it and share it amongst multiple service instances we can store that in a shared cache. This can be done by using Hazelcast shared cache or a data store like Mongo DB/SQL. Now the data model object if persisted as a HashMap, then can be read at any time later. There can be a possible scale-up for performance if the number of requests are quite high. In such cases we can have each service learn a separate model or we can have a shared model for prediction. If each instance learns a model, then the modelaccuracy and prediction results may vary.

F. Evaluation

There is an AbstractModelService that gives a template that all the MachineLearningModels must implement. This is to enforce a structure amongst all the models. The key functions of the AbstractModelService are

1) *create_model - Responsibility of creating the model and fitting it on train dataset evaluate model - evaluate self.model using the test data passed*
2) *Predict - Given an unlabeled dataset return the predicted class labels. This can be multiclass or single class.*
3) *flush_to_memory - Store the model into variable, or cache or the database. Make the model persistent*
4) *prepare_for_prediction - Takes care of pre-processing the train and the test data. Makes the dataset suitable for creating the model*

Each new model must implement this AbstractModelService.

G. Ensemble Approach

There is a class that takes the responsibility to get called upon from the controller. This class is instantiated with multiple/Single model based on the configuration. Since all the Preprocessor and the Models follow an Abstract Template pattern it becomes quite simple to call the individual prepare for prediction, predict methods of all the models. The results are then stored individually but the predictions can be returned as majority voting or as a dictionary. The models make up the keys and the values are made up by the class label predictions either by probability or label wise.

H. Design Patterns Used

1) *Delegator Pattern Iterator Pattern*
2) *Provider Pattern (Analogous to Spring Beans) Chain of Responsibility*
3) *Inversion of Control*

The requests come in via rest protocol and the client are server are totally decoupled here. There is no state between the caller (client side) and the callee (server side) when using REST. Also the machine learning data is fed to Ensemble of Classifiers and then the Ensemble returns the result. The preprocessing is handled by the Individual Model itself.

I. Deep Neural Network Model

In neural network the inputs are transitioned to the output using an activation function. There are various layers in a neural network. Our model has one input layer, one output and 2 hidden layers. These can be extended but may result in overfitting. Non-linear activation functions which are available in tensor ow are

1) *Sigmoid Function*

The maximum value for this function is 1 and the minimum value is 0. All the inputs higher than 1 and lower than 0 are capped to the limits respectively. We are predicting the probability hence the range 0 to 1. This forms the backbone of the first layer of DNN for our advertisers model.

2) Tanh Function

Range of Tanh is from -1 to 1. Strongly negative and strongly positive classification can be done. It is primary use case for 2 class classification. Since we don't have any negative classification for advertiser data we can turn down the possible usage for our model

3) Rectified Linear Unit(ReLU)

It is half rectified function.

F(x) = 0; x <= 0 (5)

F(x) = x; x > 0 (6)

Negative values are not mapped with precision here. This is used in the second layer of DeepNeuralNetwork for advertisers.

4) Softmax function

Normalizes all the input values so that the sum is 1 for all the values. This is perfect for the last layer of the DNN as we need probability of each class. The use case is not limited to binary classification. This is more suitable for multiclass classification.

FIGURE II. Sigmoid Function

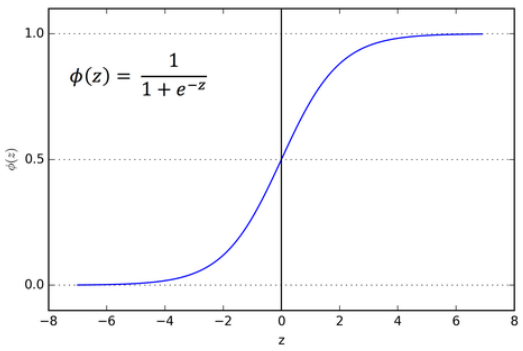


FIGURE III. Tanh vs Sigmoid Function

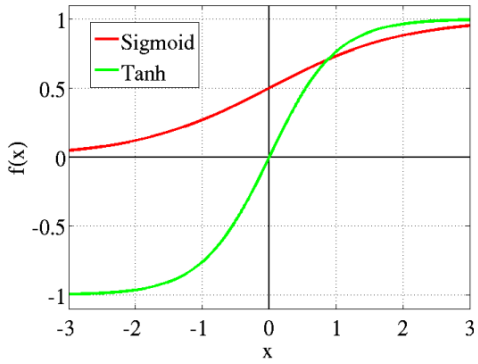
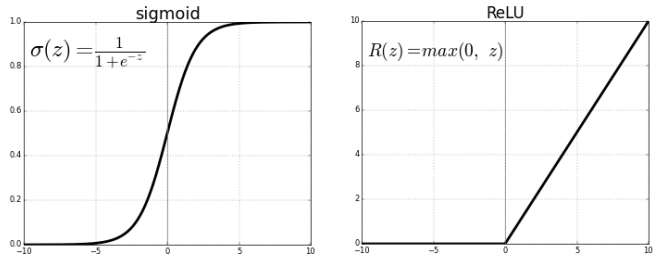


FIGURE IV. Sigmoid vs RELU function



J. The Support Vector Machine

SupportVectorMachine is trained using scikit-learn framework. The methods are

1) classifier=svm.SVC(verbose=False, probability=True)

2) classifier.fit(feature data, label data)

3) classifier.predict_proba(feature vector)

The kernel function that is used in learning the model is rbf. Other kernel parameters were also used however the linear classifier was stuck in for a very long time because the dataset is complex and cannot be divided by a linear boundary. This SupportVectorMachine also extends the AbstractMachineLearningModelService and overrides all the abstract methods. The model is trained using 90% train and 10% test dataset. The probability=True parameter states that the result obtained will be in the form of a probability distribution.

The kernel methods supported for SVM in scikit-learn are

1) rbf(Radial Basis Function)

Rbf stands for Radial Basis Kernel. This kernel is computed using the Euclidian distance between two points and a free parameter

2) poly

Polynomial kernel uses a polynomial of degree d. The dataset is converted into vectors i.e. tensor matrix. These are useful for

defining the parameters for the polynomial function. The polynomial kernel is best suited for Natural Language Processing with d=2

$F(x) = a_n x^n + a_{n-1}(x^{n-1}) + a_2 x^2 + a_1 x^1 + a_0$

3) linear

A linear function like that of y = mx+c. It is a polynomial function with d=1.

4) sigmoid

The sigmoid function is similar to that used in tensor ow Deep Neural Network model in Google's tensor ow implementation

K. Random Forest Classifier

This is a type of classifier which uses DecisionTreeClassifier at its core. It trains a number of Decision Trees. When doing so it partitions the data and trains individual trees. This in general employs ensemble like method. The end prediction is a combination of votes from the ensemble of individual Decision Trees. The predicted class is again a combination of probabilistic combination. Methods used are -

1) classifier = RandomForestClassifier ()
2) classifier.fit (feature_data, label_data)
3) classifier.predict_proba(feature vector)

L. Preprocessor Delegators

Each Model has got separate delegators for text and categorical data. This is done so that each model can have custom functionality or simply call the super method of the AbstractDataPreprocessorDelegator Now here each class has got its own Categorical and Text datapreprocessor. Each class can thus call super().delegate() or implement custom delegation process if required in future.

M. Categorical Data Preprocessors

A customized logic to convert text to categories is created. We have used python dictionaries to create the categories map. Different Categories are grouped by the feature vector.

N. Text Data Preprocessor

Herein we use the keras tokenizers. Keras tokenizers convert text data into a tensor matrix. The method used to create a tokenizer is

{

"Default": tf.keras.preprocessing.text.Tokenizer (num_words = None, filters = '', lower = True, split = '', char level = False, max_token = None, document count = 0)

}

Now, there is a provision for adding custom tokenizer per feature. If feature tokenizer is not found then we default to the "default" tokenizer. The different properties for a particular tokenizer are num words - maximum number of words to store in the dictionary. The words are sorted based on frequency of occurrences filters - remove the punctuations. A regex basically lower - convert all the text to lowercase. This is for case insensitive comparison

O. Data Preprocessor Delegator

The datapreprocessor delegator called as AbstractDataPreprocessorDelegator. This is used to delegate the preprocessing to the corresponding TextDataPreprocessor or Categorical-DataPreprocessor. This can be extended for each model differently and then the abstract method can be overridden.

III. EXPERIMENTS AND EVALUATION

There were total 4 models trained for the advertiser's dataset. These are

1) Deep Neural Network (Tensor ow Implementation)
2) Random Forest Classifier (scikit-learn Implementation)
3) Support Vector Machine (scikit-learn Implementation)
4) Ensemble Model (Tensor ow + scikit-learn Hybrid Implementation)

A. Data Shape

The dataset consisted of roughly 30000 rows. Out of these 30000 rows 3000 rows were removed for calculating the accuracy of the ensemble. The models were trained using 10% test and 25% test data samples.

B. Comparing the Accuracy of different Models for Advertisers Dataset

1) 10% Test Data Split

The accuracy achieved was 74.15% for SupportVectorMachine and 83.9% for DeepNeural-Network. The accuracy for RandomForestClassifier was very high which is about 99.9%. The ensemble of all the three models is about 83.5%. Herein the accuracy of DeepNeural-Network and Ensemble was quite comparable. The ensemble does not perform better here because the test data split was only at 10%. This suggests that the models that contribute to the ensemble must not be correlated. If there is co-relation between the models, and if a particular data tuple is misclassified, then the ensemble can also misclassify. Hence the correlation should be minimal in all of the individual classifiers. Summarizing in the table

Table 1: Accuracy Comparison for 10%

Test Data

Model	Accuracy
SupportVectorMachine	74.15%
DeepNeuralNetwork	83.9%
RandomForestClassifier	99.9%
Ensemble	83.5%

Table 2: Accuracy Comparison for 25% Test Data

Model	Accuracy
SupportVectorMachine	72.31%
DeepNeuralNetwork	83.55%
RandomForestClassifier	99.9%
Ensemble	85.1%

FIGURE V. Accuracy Comparison 10% Dataset

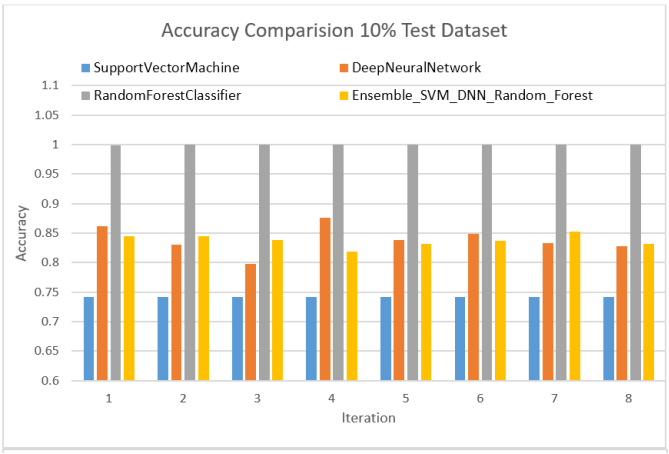
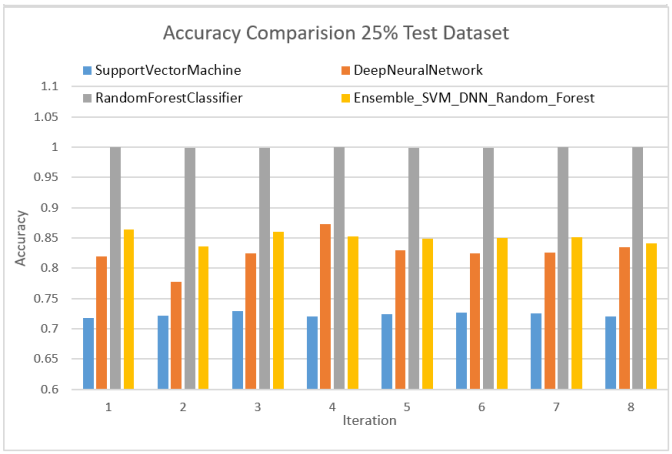


FIGURE VI. Accuracy Comparison 25% Dataset



2) 25% Test Data Split

The accuracy for the respective classifiers is listed here in above table

3) Comparing the Accuracy of different models for different datasets

We compared the ensemble of classifiers which we learnt for Campaign Advertisers with the different models that were trained for Credit Card Fraud Detection in [17]. The results are summarized in the graph. We can see that the ensemble for campaign advertisers had an accuracy of 85% approx. The Ensemble for Credit Card Fraud detection has a comparable accuracy of 84%. The neural network has the lowest accuracy for credit card dataset of around 78.9%.

We also compared the ensemble model with the models learnt for Iris dataset. The Linear Regression has the lowers accuracy of 80% for Iris Dataset. The Linear Discriminant had an accuracy of 96%. The accuracy of Naive Bayes and Decision tree were comparable to the ensemble for Campaign Advertisers [8].

FIGURE VII. Comparison of Campaign Advertisers Ensemble with Different models learnt for Credit Card Dataset

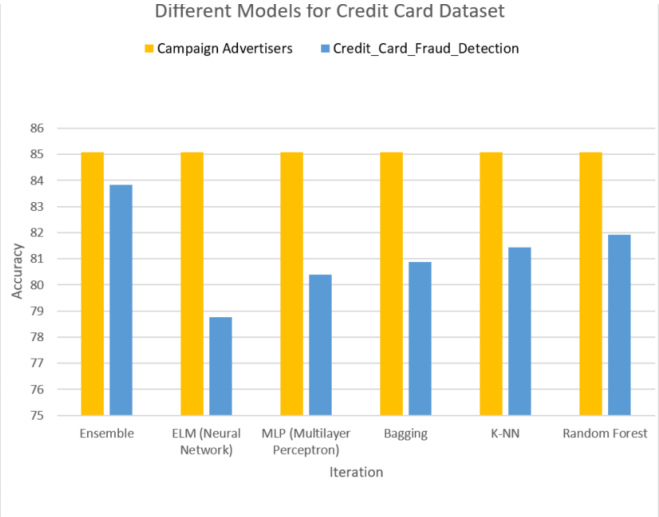
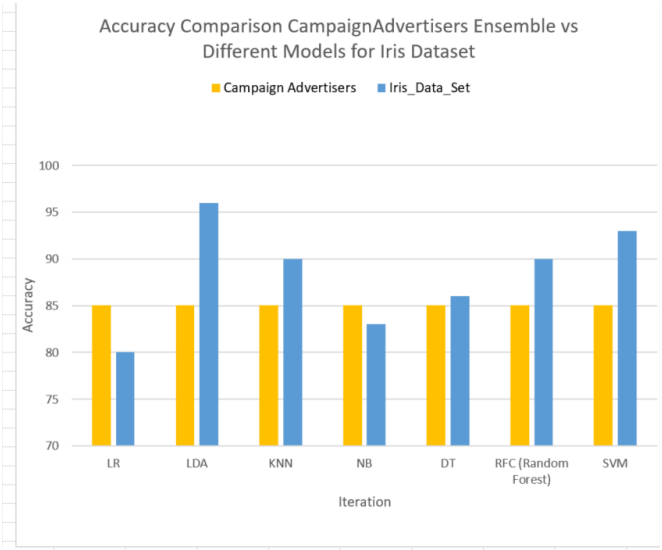


FIGURE VIII. Comparison of Campaign Advertisers Ensemble with Different models learnt for Iris Dataset



IV. CONCLUSION

We can conclude that the ensemble of classifier that is learnt for the Campaign Advertisers Dataset performs very well compared to DNN and SVM. However high accuracy was obtained using RandomForestClassifier. Since the main aim here is to create a Machine Learning model that has a more generic purpose, we have used an ensemble approach. Even if the dataset changes, we are expecting more or less similar accuracy as individual models perform differently on different datasets. Thus the ensemble performs the weighted voting for predicting the class label, thus eliminating the chances of misclassification in a continuously changing production dataset

REFERENCES

[1] Dalibor Bui, Jasminka Doba, \Lyrics classi cation using Naive Bayes", 2018 41st International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO), IEEE, Opatija, Croatia, 02 July 2018, PP 1011-1015

[2] O. Obulesu, M. Mahendra, M. ThrilokReddy, \Machine Learning Techniques and Tools: A Survey", 2018 International Conference on Inventive Research in Computing Applications (ICIRCA), IEEE, Coimbatore, India, 03 January 2019, PP 605-611

[3] Xianwei Gao, Chun Shan, Changzhen Hu, Zequn Niu, Zhen Liu, \An Adaptive En-semble Machine Learning Model for Intrusion Detection", IEEE Access(Volume 7), IEEE, 19 June 2019, PP 82512-82521

[4] Xiaobo Liu, Zhentao Liu, Guangjun Wang, Zhihua Cai, Harry Zhang, \Ensemble Transfer Learning Algorithm", IEEE Access(Volume 6), IEEE, 13 December 2017, PP 2389-2396

[5] Anjan Kumar Amirishetty, Yunrui Li, Tolga Yurek, Mahesh Girkar, Wilson Chan, Graham Ivey, Vsevolod Panteleenko, Ken Wong, \Improving Predictable Shared-Disk Clusters Performance for Database Clouds", 2017 IEEE 33rd International Conference on Data Engineering (ICDE), IEEE, San Diego, CA, USA, 18 May 2017, PP 237 -242

[6] White Paper from Oracle, \Oracle NoSQL Database for Time Series Data", December 2017

[7] Chetan Jaiswal, Vijay Kumar, \DbHAaaS: Database High Availability as a Service", 2015 11th International Conference on Signal-Image Technology and Internet-Based Systems (SITIS), IEEE, Bangkok, Thailand, 08 February 2016, PP 725-732

[8] Vasileios Tsoukas, Konstantinos Kolomvatsos, Vasileios Chioktour, Athanasios Kakaroun-tas, \A Comparative Assessment of Machine Learning Algorithms for Events Detec-tion", 2019 4th South-East Europe Design Automation, Computer Engineering, Com-puter Networks and Social Media Conference (SEEDA-CECNSM), IEEE, Piraeus, Greece, 21 November 2019, PP 1-4

[9] AJAY SHRESTHA AND AUSIF MAHMOOD, \Review of Deep Learning Algorithm-sand Architectures", IEEE Access (Volume 7), IEEE, 22 April 2019, PP 53040-53065

[10] Stelios E. Papadakis, Vangelis A. Stykas, George Mastorakis1 and Constandinos X. Mavromoustakis, \A hyper-box approach using relational databases for large scale machine learning", 2014 International Conference on Telecommunications and Multi-media (TEMU), IEEE, Heraklion, Greece, 09 October 2014, PP 69-73

[11] Hanane Bais,Mustapha Machkour, Lahcen Koutti , \Querying Database using a uni-versal Natural Language Interface Based on Machine Learning", 2016 International Conference on Information Technology for Organizations Development (IT4OD), IEEE, Fez, Morocco, 26 May 2016, PP 1-6

[12] Fatih Ertam , Galip Aydin, \Data Classi cation with Deep Learning using Tensor-ow", 2017 International Conference on Computer Science and Engineering (UBMK), IEEE, Antalya, Turkey, 02 November 2017, PP 755 - 758

[13] T. Goswami and U. B. Roy, \Predictive Model for Classi cation of Power System Faults using Machine Learning," TENCON 2019 - 2019 IEEE Region 10 Conference (TENCON), Kochi, India, 2019, pp. 1881-1885

[14] I. Stanin and A. Jovi, "An overview and comparison of free Python libraries for data mining and big data analysis," 2019 42nd International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO), Opatija, Croatia, 2019, pp. 977-982

[15] H. Suen, K. Hung and C. Lin, "TensorFlow-Based Automatic Personality Recognition Used in Asynchronous Video Interviews," in IEEE Access, vol. 7, pp. 61018-61023, 2019

[16] N. Ari and M. Ustazhanov, "Matplotlib in python," 2014 11th International Confer-ence on Electronics, Computer and Computation (ICECCO), Abuja, 2014, pp. 1-6

[17] D. Prusti and S. K. Rath, "Fraudulent Transaction Detection in Credit Card by Ap-plying Ensemble Machine Learning techniques," 2019 10th International Conference on Computing, Communication and Networking Technologies (ICCCNT), Kanpur, India, 2019, pp. 1-6

S.R.Khonde - working as Assistant Professor in Computer Engineering at M. E. S. College of Engineering Pune India from 2006. She obtained bachelor's degree in computer engineering from North Maharashtra University, Jalgaon in May 2004 & master's degree in computer engineering in December 2011 from University of Pune. She is currently pursuing PhD in Computer Engineering from Sathyabama Institute of Science and Technology, Chennai, India. She published a number of papers in international journals and conferences. She has more than 14 years of teaching. Her area of interest is data mining, information and network security. She is life member of Indian society ISTE from 2005, IET and CSI.