TikTok Commercial Video Advertisements Detect!

Who Can Make The Accuracy 0.99?

Capstone Project 2

By

Vertulie Pierre Louis

Why is it a useful question to answer and details about the client? TikTok is the leading destination for short-form mobile video. I am interested in learning about TikTok’s advertisement.

Purpose, motivation and description:

TikTok commercial video advertisements detect! Who can make the accuracy 0.99?

The purpose of the project is to predict the label and make the accuracy 0.99?

For the prediction, we used Random Forest as the base model to see how good it will perform. Including Neural network (sklearn's MLPClassifier) with basic 5 layers and this setting of neurons: (16,32,64,32,16) and the keras to build our models. We started with a base network with 6 layers.we added dropout, more layers, changing the number of neurons to make the model more complex. We are also trying different activation functions- relu.

For the Split test, the dataset has been separated into training and Testing data so that prediction can be done on the testing data. I also import Tableau that can make the visualization more feasible. The technical aspects used for visualization are Matplolib, seaborn Including data manipulation: pandas, NumPy, modeling-Sklearn, Neural network & Keras.

Variable/Feature

My target label is -1 to 0. It is an integer.

Data acquisition

The dataset for this project is downloaded as a csv file from the Kaggle website.
https://www.kaggle.com/lilhxr/commercial-vedio-data

Data Management

The dataset contains 29685 rows and 232 columns. It is in csv file.

The dataset provides information about 129685 videos that are stored in the commercialvediodata.com

Data Wrangling & Cleaning

The goal of the wrangling step is to transform the data from csv format into a pandas dataframe.

When I took a closer look at the data, I noticed that several features have Null values. I performed **data wrangling** using df.isnull() a method that returns True if there is any NaN value in the DataFrame and returns False if there isn’t any.

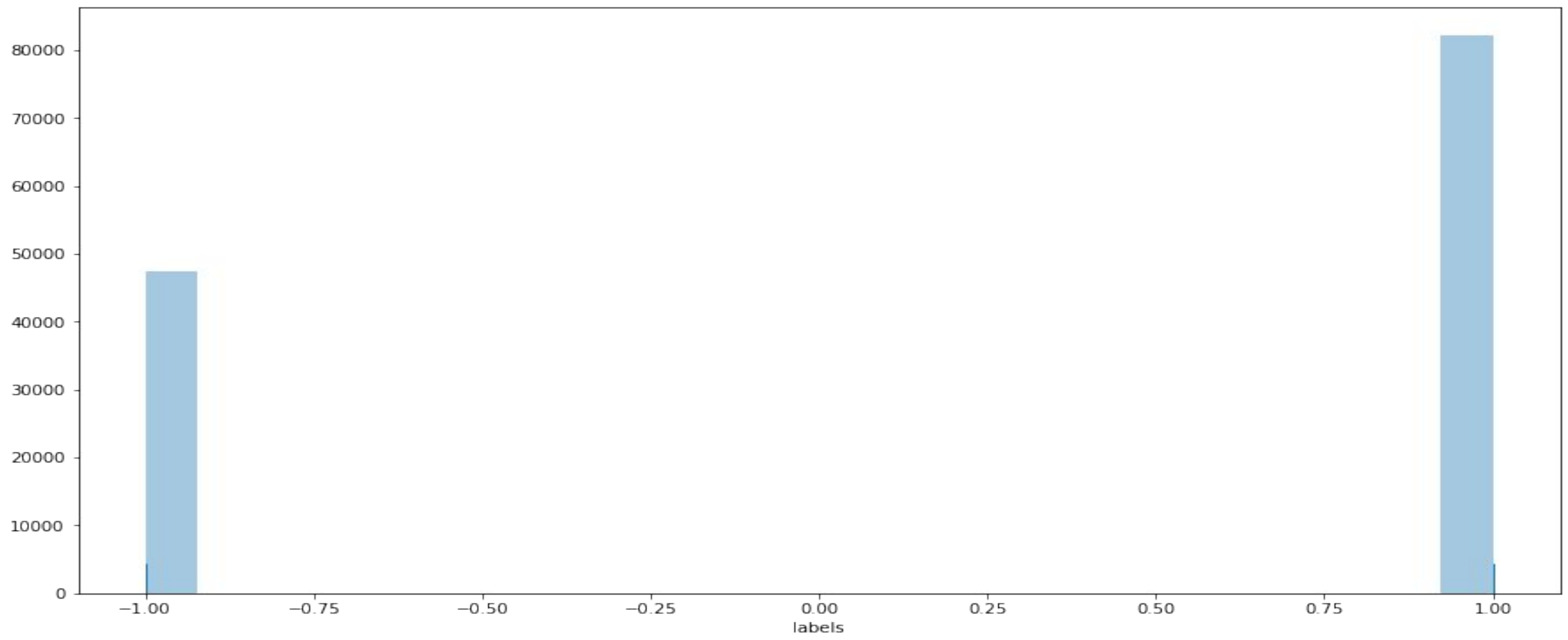
I also noticed that there was a mixture of True or False in the data then I used the fillna(df.mean()) function to filling all the null values.

Exploratory Data Analysis Summary/Visualization

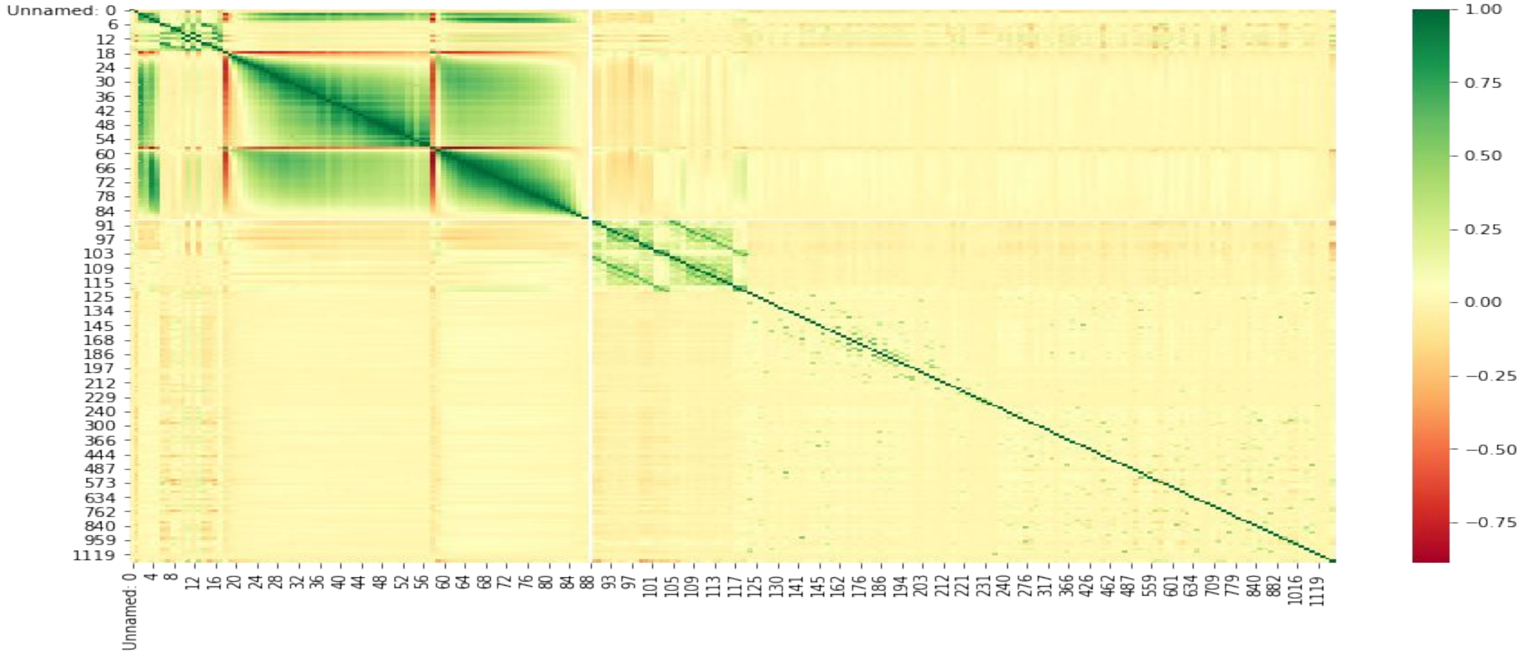
I used the describe() function to get a basic statistical summary about the data such as: Percentile, Mean, Median, Variance, Std and to determine anomaly in the data.

The visualization is presented by a bar plot and correlation plot. The bar plot makes it easy to compare the data between the labels and the other variables in the dataset. From the data when the label is 1.0 there are 88000 zeroes. When the label is -1 there are 45000. We used the Correlation to assess a possible linear association between the labeled and the other variables in the data.

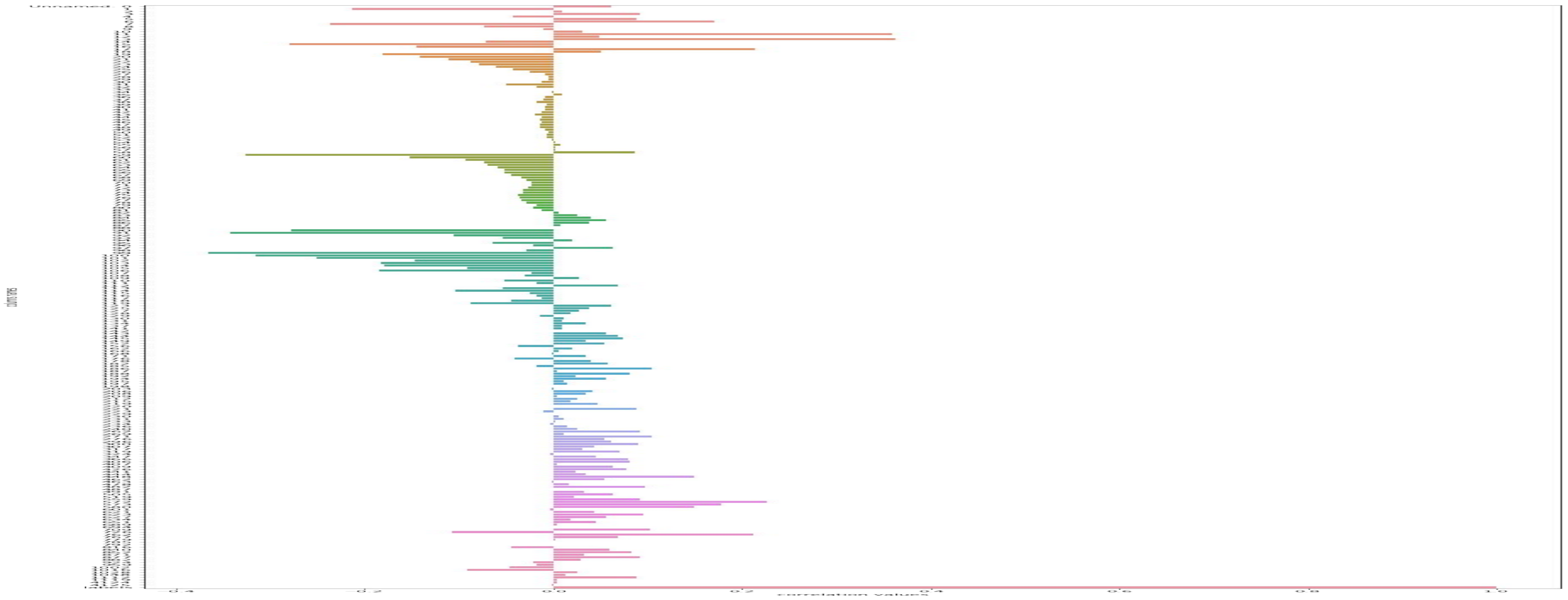
From the data When the label is 1.0 there are 88000 zeroes. When the label is -1 there are 45000



Now we plot the correlation matrix to see how variables are correlated with the target variable. We can see some of features are dependent on each other.



This graph shows a relationship between correlation value and column name.



Data Normalization/Preprocessing

we scaled the data using the standard scaling method.

Splitting The Data Into Training And Test Sets

To assess the model’s performance, I divided the data set into two parts: a training set and a test set. The first is used to train the system, while the second is used to evaluate the learned or trained system.

Feature Selection:

We dropped some unnecessary columns from the dataset after the Pearson correlation analysis. X=df.drop("labels",axis=1

This column did not have any effect on the output.

Modeling:

RANDOM FOREST

We tried Random Forest as the base model to see how good it is performed. it performed quite well and resulted in 95% accuracy on the test set.

Advantages

The random forest algorithm works well when the data sets have both categorical and numerical features. The random forest algorithm also works well when data has missing values or it has not been scaled

Disadvantages

A major disadvantage of random forests lies in their complexity. Random Forest required much more computational resources, owing to the large number of decision trees joined together. Due to their complexity, they require much more time to train than other comparable algorithms.

Multi-Layer Perceptron (MLP) Classifier Algorithm¶

I used MLPClassifier to implement a multi-layer perceptron (MLP) algorithm that trains the model using Backpropagation

Multi-layer Perceptron (MLP)

The advantages of using Multi-layer Perceptron are:

- Capability to learn non-linear models.
- Capability to learn models in real-time (on-line learning) using partial_fit).

The disadvantages of Multi-layer Perceptron (MLP) include:

- MLP with hidden layers have a non-convex loss function where there exists more than one local minimum. Therefore different random weight initializations can lead to different validation accuracy.
- MLP requires tuning a number of hyperparameters such as the number of hidden neurons, layers, and iterations.
- MLP is sensitive to feature scaling.

Using keras

Now we are trying the keras to build our own neural network. We started with a base network with 6 layers. Now we are trying adding dropout, adding more layers, changing the number of neurons to make the model more complex. We are also trying different activation functions- relu.

How Do We Picked Parameters For The Model Used?

For Random Forest I used: max depth and n_estimators

Multi-Layer Perceptron (MLP) Classifier
hidden__layer__sizes : This parameter allows us to set the number of layers and the number of nodes we wish to have in the Neural Network Classifier.

- max__iter: It denotes the number of epochs.
- activation: The activation function for the hidden layers.
- solver: This parameter specifies the algorithm for weight optimization across the nodes.
- random__state: The parameter allows to set a seed for reproducing the same results

Model Building

To building the multi-layer perceptron we use the Keras Sequential model: it’s a linear stack of layers. We create the model by passing a list of layer instances to the constructor, which we set up by running `model = Sequential()`.

For the input shape: I used `input_dim`

Dense layers implement the following operation: `output = activation(dot(input, kernel) + bias)`

In the first layer, the `activation` argument takes the value `relu`

We also add dropout, adding more layers, changing the number of neurons to make the model more complex.

Communicate Findings

The model aims to predict the accuracy of 0.99. For the prediction, We tried Random Forest as the base model to see how good it is performed. it performed quite well and resulted in 95% accuracy on the test set.

We also use keras to build our neural network. We started with a base network with 6 layers. To make the model more complex; we included dropout, more layers, changing the number of neurons and using different activation functions- relu This model also performed well with an accuracy of 0.9362