Predicting Customer Ad Click using Machine Learning

Kunal Panjwani (202018001), Vanditha Vinod (202018003), Nihar Shah (202018014), Yagn Purohit (202018035)

Abstract—In this project, we will predict whether a person will click on an ad targeted to him/her. We have used various classifiers in this model to reach an optimal accuracy score. The classifiers we used are Logistic Regression, Decision Tree Classifier, Support Vector Machine and Random Forest Classifier. We then used Voting Classifier to train the dataset on these models. At last, we used KNN classifier model and compared the results we obtained and concluded the best model.

1 Introduction

Advertising plays a vital role in spreading awareness around the world. It can significantly help a business to flourish, educate people and keep them informed. In this age of advanced technology, it is ideal to advertise on the internet where it can reach a huge audience. People with niche businesses can find the target audience, which is not easy, if not for online advertisements. It is necessary to target the ads to the appropriate audience, and this is where this project comes to use. Online advertisement has made it possible to reach the targeted audience globally and not just in a local area.

1.1 Dataset

In this project, we have taken an advertising dataset from Kaggle[1], which has various features of the users like:

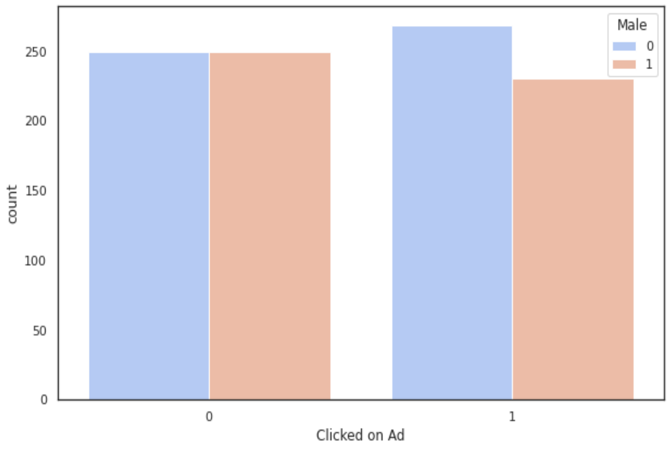
- Daily Time Spent on Site
- Age
- Daily Internet Usage
- Area income
- Ad topic line
- Gender
- City
- Country
- Timestamp

The dataset has 9 features and 1000 observations, each observation representing a user. We split this data into training and testing data.

The "Clicked on Ad" column has to be predicted for the testing data.

The labels we have to predict are in binary form 0 or 1 (0: The user did not click on the ad, 1: The user clicked on the ad). Clearly, this is a binary classification problem, and using the given features, we have to predict whether a user clicked on an ad or not.

We pre-processed the data and found various columns like "Ad Topic Line", "Country" and "City" were providing negligible contribution in predicting the labels, and hence we dropped them. We also split the "Timestamp" column into "Day of the month", "Day of the week", "Month" and "Hour" columns for better understanding.



On visualization, we observed that "Daily less internet usage" users tend to click on ads more. Also, females are more likely to click on ads as compared to males. In the dataset, most users are in the age group of around 30 years, but we observed that most of the people who clicked on ads were around the 40 years' age group.

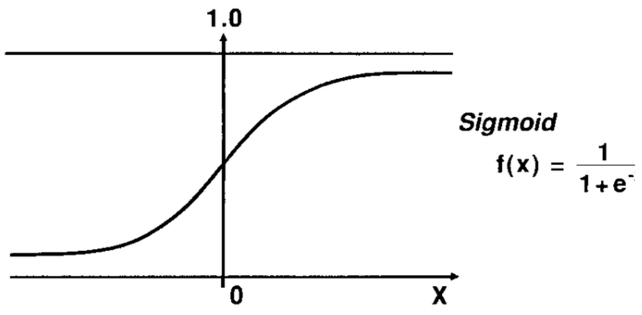
2 Methods

As the dataset represents a binary classification problem, we have applied various classification algorithms using the standard sci-kit learn packages[2]. Classification algorithms in machine learning use input training data to predict the likelihood that subsequent data will fall into one of the predetermined categories.

In our project, these predetermined categories are whether a customer will click on the ad or not (Stored as 1 or 0 respectively).

2.1 Logistic Regression

Logistic Regression[3] can be applied to binary or more classification problems. It uses a logistic function which is a sigmoid function given by:



which takes real input x and gives output between 0 and 1.

Logistic regression is easier to implement, interpret, and very efficient to train, and also logistic regression is better than linear regression because the sigmoid curve fits better to the data than a straight line. But the major limitation of Logistic Regression is the assumption of linearity between the dependent variable and the independent variables. In other words, non-linear problems can't be solved with logistic regression because it has a linear decision surface

2.2 Decision Tree Classifier

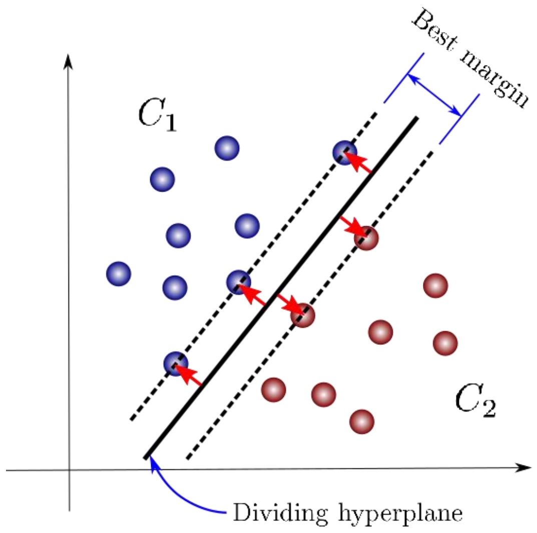
A Decision Tree[4][10] is a simple representation for classifying examples. It is Supervised Machine Learning where the data is continuously split according to a certain parameter.

Decision tree obtained in our project.

Compared to other algorithms, decision trees require less effort for data preparation during pre-processing. A decision tree does not require normalization or scaling of the data. Also, a decision tree model is very intuitive and easy to explain. But, decision tree training is relatively expensive as the time taken and complexity are more.

2.3 SVM

The Support Vector Machine (SVM)[5] is a linear classifier that can be viewed as an extension of the Perceptron algorithm. The Perceptron guarantees that you find a hyperplane if it exists. The SVM maximizes the margin separating hyperplane. SVM works relatively well when there is a clear margin of separation between classes. SVM is more effective in high-dimensional spaces. Also, SVM is relatively memory efficient. SVM does not perform very well when the data set has more noise i.e. target classes are overlapping. As the support vector classifier works by putting data points, above and below the classifying hyperplane there is no probabilistic explanation for the classification.



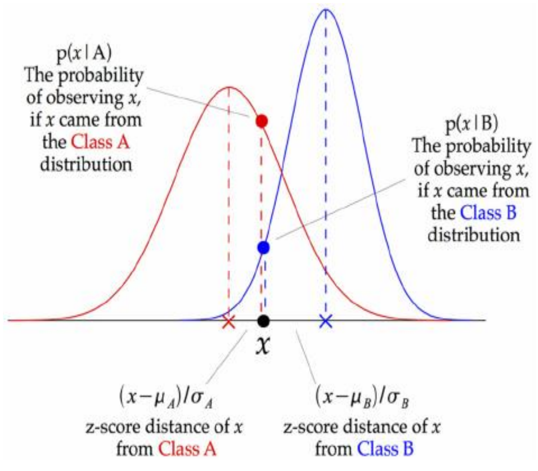
2.4 Random Forest

Random forest[[6][11] is a supervised learning algorithm, which is used for both, classification as well as regression. But however, it is mainly used for classification problems. Random forest algorithm creates decision trees on data samples and then gets the prediction from each of them and finally selects the best solution by means of voting. It is an ensemble method, which is better than a single decision tree because it reduces the over-fitting by averaging the result. Just like the decision tree classification algorithm, the Random Forest algorithm may change considerably by a small change in the data. Random Forest algorithm computations may go far more complex compared to other algorithms.

A decision tree from the random forest obtained in our project.

2.5 Gaussian Naïve Bayes Classification

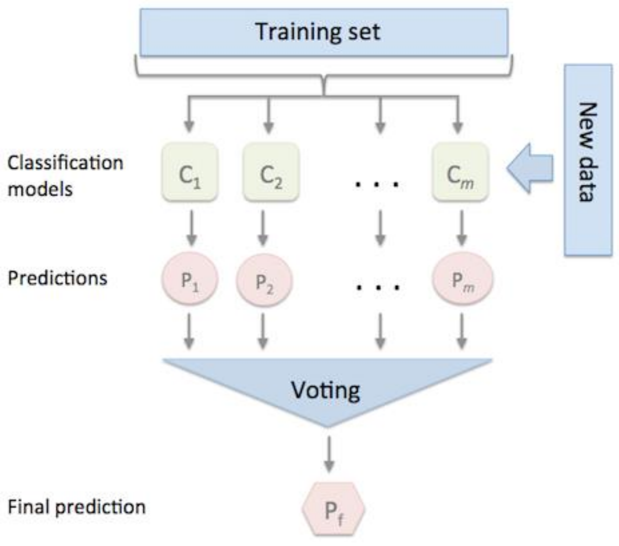
The Naive Bayes classifier[7][12] separates data into different classes according to the Bayes' Theorem, along with the assumption that all the predictors are independent of one another. It assumes that a particular feature in a class is not related to the presence of other features. This algorithm works very fast and can easily predict the class of a test dataset. Naive Bayes classifier performs better than other models with less training data if the assumption of independence of features holds. It assumes that all the features are independent. While it might sound great in theory, in real life, you'll hardly find a set of independent features. If the predictors aren't discrete but have a continuous value, we assume that they are a sample from a gaussian distribution.



p(x | A)
The probability of observing x, if x came from the Class A distribution
p(x | B)
The probability of observing x, if x came from the Class B distribution
x
$(x-\mu_A)/\sigma_A$
z-score distance of x from Class A
$(x-\mu_B)/\sigma_B$
z-score distance of x from Class B

2.6 Voting Classifier

A Voting Classifier[8][13] is a machine learning model that trains on an ensemble of numerous models and predicts an output (class) based on their highest probability of chosen class as the output. One of the major advantages to using a voting classifier is it reduces the error and reduces variance. Also, Voting is one of the simplest ways of combining the predictions from multiple machine learning algorithms. Voting classifier isn't an actual classifier but a wrapper for a set of different ones that are trained and evaluated in parallel in order to exploit the different peculiarities of each algorithm.



In our project, all the above classification algorithms are used in this voting classifier which has, predictably so, given the best accuracy among these algorithms.

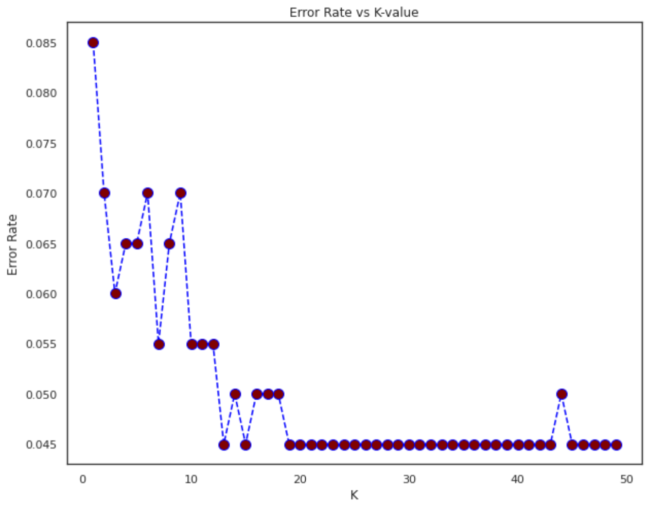
2.7 KNN (k - nearest neighbors) Classifier

K-nearest neighbors (KNN)[9][14] algorithm uses 'feature similarity' to predict the values of new data points which further means that the new data point will be assigned a value based on how closely it matches the points in the training set.

While applying this algorithm the most important thing to keep in mind is to scale all the values properly which is what we have done for our project.

KNN's main disadvantage of becoming significantly slower as the volume of data increases makes it an impractical choice in environments where predictions need to be made rapidly. Moreover, there are faster algorithms that can produce more accurate classification and regression results. However, provided you have sufficient computing resources to speedily handle the data you are using to make predictions, KNN can still be useful in solving problems that have solutions that depend on identifying similar objects. An example of this is using the KNN algorithm in recommender systems, an application of KNN-search.

But it does not work well with large dataset as calculating distances between each data instance would be very costly. Also, does not work well with high dimensionality as this will complicate the distance calculating process to calculate distance for each dimension.



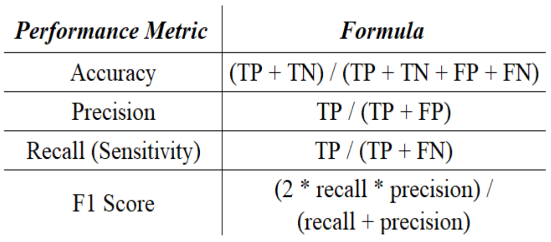
So coming to the question on how to select the "k" value, we ran a for loop from 1 to 50 and predicted y-values and then calculated the error for each iteration and plotted k vs error graph. By doing so, we concluded that 20 to 40 is giving a constant low error and thus we went with k=20.

3 Results

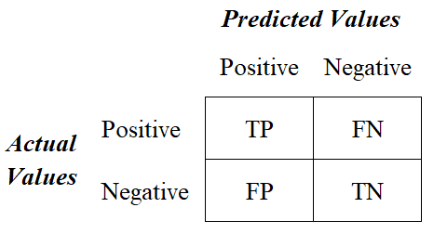
After performing the methods mentioned above, these are the results that were obtained:

Method	**Accuracy Score**	**Precision**	**Recall**	**F1 Score**
Logistic Regression	0.9060	0.91	0.91	0.91
Decision Tree Classifier	0.9363	0.94	0.94	0.94
SVM	0.9696	0.97	0.97	0.97
Random Forest Classifier	0.9454	0.95	0.95	0.95
Gaussian Naive Bayes	0.9606	0.96	0.96	0.96
Voting Classifier (Hard)	0.9750	0.98	0.97	0.97
Voting Classifier (Soft)	0.9545	0.95	0.95	0.95
KNN	0.9550	0.96	0.95	0.95
Maximum	Voting Classifier (Hard)	Voting Classifier (Hard)	SVM and Voting Classifier (Hard)	SVM and Voting Classifier (Hard)

The above metrics are calculated by:



Performance Metric	***Formula***
Accuracy	(TP + TN) / (TP + TN + FP + FN)
Precision	TP / (TP + FP)
Recall (Sensitivity)	TP / (TP + FN)
F1 Score	(2 * recall * precision) / (recall + precision)



		Predicted Values	
		Positive	Negative
Actual Values	Positive	TP	FN
	Negative	FP	TN

4 Conclusion

In this project, we have performed five classification algorithms and then applied voting classifier (both hard and soft) using them. We also performed KNN classifier algorithm.

From the table shown above, an ensemble of the first five algorithms, i.e. voting classifier gives the best accuracy, precision, recall and F1 score and SVM gives maximum recall and F1 score.

We have obtained fairly accurate predictions on whether a person will click on an ad or not. This can be used by many companies/organizations who want to advertise themselves/ their products globaly.

5 References

[1] Dataset- https://www.kaggle.com/debdyutidas/advertisingcsv

[2] Scikit learn for machine leaarning- https://scikit-learn.org/stable/

[3] Logistic Regression (sklearn)- https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LogisticRegression.html

[4] Decision tree (sklearn)- https://scikit-learn.org/stable/modules/generated/sklearn.tree.DecisionTreeClassifier.html

[5] SVM (sklearn)- https://scikit-learn.org/stable/modules/generated/sklearn.svm.SVC.html

[6] Random forest (sklearn)- https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html

[7] Gaussian Naïve Bayes (sklearn)- https://scikit-learn.org/stable/modules/generated/sklearn.naive_bayes.GaussianNB.html

[8] Voting Classifier (sklearn)- https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.VotingClassifier.html

[9] KNN (sklearn)- https://scikit-learn.org/stable/modules/generated/sklearn.neighbors.KNeighborsClassifier.html

[10] Decision tree- https://towardsdatascience.com/understanding-decision-tree-classifier-7366224e033b

[11] Random Forest- https://towardsdatascience.com/understanding-random-forest-58381e0602d2

[12] Gaussian Naïve Bayes- https://towardsdatascience.com/learning-by-implementing-gaussian-naive-bayes-3f0e3d2c01b2

[13] Voting classifier- https://medium.com/@sanchitamangale12/voting-classifier-1be10db6d7a5

[14] KNN- https://towardsdatascience.com/machine-learning-basics-with-the-k-nearest-neighbors-algorithm-6a6e71d01761