Individual Project

Aleksandr Shkurin

The goal of this project is to demonstrate the author's knowledge of how learning machines work. Knowledge will be demonstrated by applying five machine learning models on the dataset specified by the project conditions. In the course of the project itself, the techniques of preprocessing on the database, the choice of functions and parameters, the use of various systems for evaluating the performance of machine learning models, and other methods were applied.

In this document, the talk will go about the process of machine learning in general, as well as a description of all the steps that were taken during the project. First, the topic will be about the general theory of machine learning, then about working with a dataset, and then we will sequentially describe how various theories and steps of machine learning were applied in practice. Also, it is worth mentioning that the whole project was done in R programming language.

Introduction

A subset of artificial intelligence (AI), machine learning (ML) is a field of computational science that focuses on the analysis and interpretation of patterns and structures in data to enable learning, reasoning, and decision making outside of human interaction. Simply put, machine learning allows the user to feed a huge amount of data to a computer algorithm, and the computer analyzes and makes data-driven recommendations and decisions based only on the input. If any corrections are identified, the algorithm may include this information to improve future decision making.

Machine learning has three parts:

- The computational algorithm which enforces decision making.
- Variables or features and parameters influencing the decision.
- Basic knowledge for what the actual answer is, which allows the system to learn.

Initially, parameter data for which the answer is known is fed into the model. The algorithm is then run and adjusted until the result of the algorithm (training) is consistent with the known answer. At this stage, increasingly large amounts of data are introduced to help the system learn and process more complex computational decisions.

Data is the blood of any business. Data-driven decisions are increasingly making the difference between keeping up with the competition or falling further behind. Machine learning can be the key to unlocking the value of corporate and customer data and making decisions that keep a company ahead of the competition.

In our case, the data (*bank_mkt_train.csv*) is financial or banking information about customers. In this case, the target variable in our data is a binary column that determines if the user has subscribed to a particular service or not.

In general, the use of bank data is one of the most powerful ways to benefit from the use of machine learning. Automated solutions with big data capabilities can track and store as much customer information as needed, providing the most accurate and personalized customer experience. Having a variety of information about user behavior allows financial companies to know what customers currently want, as well as what they are willing and able to pay for.

Specifically, in our example, we have a database with twenty-one columns that describes the historical data about the client and his financial condition. We are presented with columns with information such as the client's age (*age*), his marital status (*marital*), his current job (*job*), the client's education (*education*), and other data that relates to financial information like employment variation rate (*emp.var.rate*), a consumer price index (*cons.price.idx*), a consumer confidence index (*cons.conf.idx*) and so on. Based on these data, models will be compiled and applied.

Data preprocessing

Data pre-processing is a data mining technique that turns raw data collected from various sources into cleaner information that is more workable. In other words, it is a preliminary step in which all available information is used to organize, sort and combine it.

The raw data may have missed or inconsistent values and may contain a lot of redundant information. The most common problems with raw data can be divided into 3 groups:

- Missing data, which also can be considered as inaccurate data, as information that is not present, which creates gaps that may be relevant to the final analysis.
- Noisy data, that includes erroneous data and outliers that you can find in the dataset, but it's just nonsense information.
- Inconsistent data, which occur when you store files with similar data in different formats and files.

Also, data must be in a format appropriate for the machine learning algorithm. For example, if the data contains columns with the text values and the algorithm requires only the data in the numerical format, then the categorical variables treatment is needed. In our case that was the first point to do.

Variables can be characterized as quantitative or qualitative (also known as categorical). Quantitative variables take numeric values. Examples include a person's age, height or income, home value and share price. In contrast, categorical variables take values in one of K different classes or categories. Examples of qualitative class variables include the gender of the person (male or female), the brand of product purchased (brand A, B, or C), the person's default on debt.

What is important to mention here is the fact that categorical variables can either have two levels or multiple levels. If a categorical predictor has only two levels, or perhaps values, it is very easy to include it in the regression model. We align just create an indicator or dummy variable that takes two possible dummy values. For example, with gender, we can represent it like this:

$$x^i = \begin{cases} 1 \text{ if } i_{th} \text{ person is female} \\ 0 \text{ if } i_{th} \text{ person is male} \end{cases}$$

When a categorical predictor has more than two levels, one dummy variable cannot represent all possible values. In this situation, we can create additional dummy variables.

In our case we have ten categorical variables and only one of the has two levels. It is *contact* which has following values: *cellular* and *telephone*. Other variables have three and more different unique values.

A data.frame: 20000 × 21

client_id	age	job	marital	education	default	housing	loan	contact	month	⋯	campaign	pdays	previous	poutcome
<int>	<int>	<chr>	<chr>	<chr>	<chr>	<chr>	<chr>	<chr>	<chr>	⋯	<int>	<int>	<int>	<chr>
29925	42	management	married	basic.9y	no	no	no	cellular	jul	⋯	1	999	0	nonexistent
37529	35	unemployed	married	university.degree	no	yes	no	telephone	jun	⋯	4	999	0	nonexistent
2757	44	technician	married	basic.9y	no	yes	yes	cellular	may	⋯	1	999	0	nonexistent
9642	45	services	married	high.school	no	yes	no	cellular	apr	⋯	1	999	0	nonexistent
14183	45	unknown	married	unknown	unknown	unknown	unknown	telephone	may	⋯	1	999	0	nonexistent
15180	38	technician	married	professional.course	no	no	no	telephone	may	⋯	2	999	0	nonexistent
27168	33	technician	married	professional.course	no	no	yes	cellular	apr	⋯	1	NA	1	failure
9097	38	blue-collar	single	basic.9y	unknown	yes	no	telephone	may	⋯	1	999	0	nonexistent

Figure 1. Original dataset

Before dummy treating, the original dataset looked as on the Figure 1. It is possible to see that all categorical variables have their text values inside of the columns.

A data.frame: 6 × 64

	client_id	age	campaign	pdays	previous	emp.var.rate	cons.price.idx	cons.conf.idx	euribor3m	nr.employed	⋯	month_sep	month_NA	day_of_week
	<int>	<int>	<int>	<int>	<int>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	⋯	<int>	<int>	
1	29925	42	1	999	0	1.4	93.918	-42.7	4.968	5228.1	⋯	0	0	
2	37529	35	4	999	0	1.4	94.465	-41.8	4.960	5228.1	⋯	0	0	
3	2757	44	1	999	0	-1.8	92.893	-46.2	1.264	5099.1	⋯	0	0	
4	9642	45	1	999	0	-1.8	93.075	-47.1	1.453	5099.1	⋯	0	0	
5	14183	45	1	999	0	1.1	93.994	-36.4	4.859	5191.0	⋯	0	0	
6	15180	38	2	999	0	1.1	93.994	-36.4	4.858	5191.0	⋯	0	0	

Figure 2. Dataset after dummy treating

After applying *dummy* function from the *dummies* library, the dataset set changed to one that is possible to see on the Figure 2. The mentioned above function returns a matrix with the number of rows equal to the that of given variable. Row names are retained if the supplied variable has associate row names. The columns are return in the same order as the input with dummy variable columns replacing the original column. It possible to notice that the number of columns changed from 21 to 64, including all new dummy variables.

The next step in data preprocessing was to remove the NA values. This step, together with the dummy encoding is important since the feature selection algorithms and models that will be applied later don't support datasets with text or NA values. Depending on the case, the treatment of NA values is usually related to either replacing them with an average value or removing at all.

```
# Check NA
apply(is.na(data), 2, sum)
```

client_id: 0 **age:** 202 **campaign:** 203 **pdays:** 185 **previous:** 209 **emp.var.rate:** 165 **cons.price.idx:** 181 **cons.conf.idx:** 197 **euribor3m:** 204 **nr.employed:** 184 **subscribe:** 0 **job_blue-collar:** 0 **job_entrepreneur:** 0 **job_housemaid:** 0 **job_management:** 0 **job_retired:** 0 **job_self-employed:** 0 **job_services:** 0 **job_student:** 0 **job_technician:** 0 **job_unemployed:** 0 **job_unknown:** 0 **job_NA:** 0 **marital_married:** 0 **marital_single:** 0 **marital_unknown:** 0 **marital_NA:** 0 **education_basic.6y:** 0 **education_basic.9y:** 0 **education_high.school:** 0 **education_illiterate:** 0 **education_professional.course:** 0 **education_university.degree:** 0 **education_unknown:** 0 **education_NA:** 0 **default_unknown:** 0 **default_yes:** 0 **default_NA:** 0 **housing_unknown:** 0 **housing_yes:** 0 **housing_NA:** 0 **loan_unknown:** 0 **loan_yes:** 0 **loan_NA:** 0 **contact_telephone:** 0 **contact_NA:** 0 **month_aug:** 0 **month_dec:** 0 **month_jul:** 0 **month_jun:** 0 **month_mar:** 0 **month_may:** 0 **month_nov:** 0 **month_oct:** 0 **month_sep:** 0 **month_NA:** 0 **day_of_week_mon:** 0 **day_of_week_thu:** 0 **day_of_week_tue:** 0 **day_of_week_wed:** 0 **day_of_week_NA:** 0 **poutcome_nonexistent:** 0 **poutcome_success:** 0 **poutcome_NA:** 0

Figure 3. Checking NAs

In order to understand which method to apply, the first step is to check the columns and the amount of NA values, as it is possible to see on the Figure 3. Since the overall amount of the NA values is not significant and that there are such values as for example campaign, which can't get the average, since it is an id of the campaign that was applied, it was decided to drop the NA values. This decreased the number of rows from 20000 to 18341, which should not affect the final prediction result.

The last step, which was applied for the data preprocessing, was the treatment of outliers. Identification of potential outliers is important for the following reasons. An outlier may indicate bad data. For example, the data may have been coded incorrectly or an experiment may not have been run correctly. Also, for example in case of regression, an outlier will increase most components of the central tendency and variance, including the mean, standard deviation, and variance. When, say, the mean and standard deviation increase, the value of T-stat increases, and this, in turn, can cause the value of p to exceed 0.5. This means that a predictor that is a significant predictor may turn out to be a non-significant predictor due to noise in the data resulting in a higher standard deviation, implying mis-modeling and therefore mismatched prediction. Due to this reason it usually recommended to treat outliers before the feature selection, but it is usually up to the decision of the data scientist for the specific case.

For example, with our case, it was decided to treat outliers after the feature selection since after analyzing the outliers it was clear that there is no significant amount of them. Also, trying to put outliers' treatment in the beginning didn't give significant difference in the sleeted features. Z-score was chosen as a method of treating outliers. Z-scores are the number of standard deviations above and below the mean that each value falls. For example, a Z-score of 2 indicates that the observation is two standard

deviations above the mean, and a Z-score of -2 indicates that the observation is two standard deviations below the mean. A Z-score of zero is a value equal to the mean. To calculate the Z-score for an observation, take the raw measurement, subtract the mean, and divide by the standard deviation. After applying Z-scores the amount of record changed from 18341 to 17472.

Feature selection

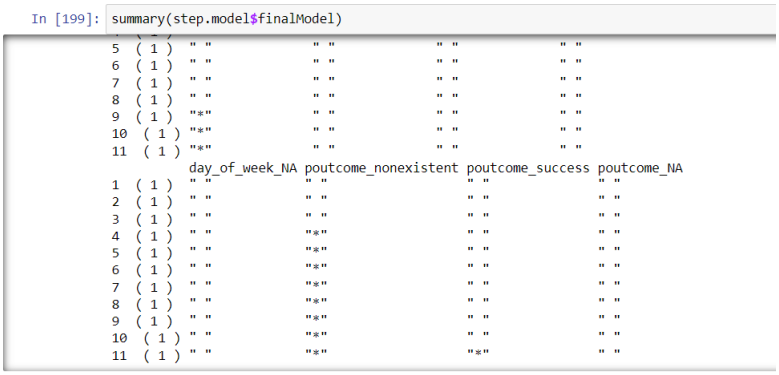
Feature selection, also known as variable selection, is the practice of selecting a subset of appropriate features, predictors, and variables for use in model building. It is an automatic selection of the attributes present in the data that are most important and relevant to the predictive modeling problem you are working on.

Removing irrelevant data improves training accuracy, reduces computation time, and improves understanding of the training model or data. When developing a machine learning model in real life, more often than not, not all variables in a dataset are useful. Adding redundant variables reduces the ability of the model to generalize and may also reduce the overall accuracy of the classifier. Also, if more variables are added to the model, it results in the development of a complex model.

For this project it was decided to use a combined Stepwise feature selection. Two main types of Stepwise selection model are forward Stepwise and backward Stepwise. Forward selection starts when there are no predictors in the model, predictors with the largest contribution are iteratively added, and stops when the improvement is no longer statistically significant. Backward selection starts with all predictors in the model, iteratively removes the least contributing predictors, and stops when you have a model where all predictors are statistically significant. The hybrid or combined Stepwise regression or stepwise selection consists of iteratively adding and removing predictors in a prediction model to find a subset of variables in the dataset, resulting in a model with the best performance, that is a model that reduces the prediction error. The methodology consists of starting with no predictors and then gradually add the most useful predictors. After each new variable is added, remove any variables that no longer improve model fit. Thus, it usually gives the best final result.

In order to implement this method, the *train* function from *caret* library was used. The *train* function provides a simple workflow for performing step-by-step selections using transitions and MASS packages. It has an option named *method* which takes as value a type of the Stepwise method that you want to use. It is also needed to specify the *nvmax* tuning parameter, which corresponds to the maximum number of predictors that should be included in the model. For this case the number of parameters was decided to set up to 11. Also, with the *trControl* parameter you need to specify the validation method, which will be used for your feature selection. After using the model, in order to display the best tuning values, *nvmax*, automatically selected by the *train* function, you need to write:

step.model$bestTune



```
In [199]: summary(step.model$finalModel)
```

```
5  ( 1 )  " "                " "                " "                " "
6  ( 1 )  " "                " "                " "                " "
7  ( 1 )  " "                " "                " "                " "
8  ( 1 )  " "                " "                " "                " "
9  ( 1 )  "*"                " "                " "                " "
10  ( 1 ) "*"                " "                " "                " "
11  ( 1 ) "*"                " "                " "                " "
          day_of_week_NA poutcome_nonexistent poutcome_success poutcome_NA
1  ( 1 )  " "              " "                 " "              " "
2  ( 1 )  " "              " "                 " "              " "
3  ( 1 )  " "              " "                 " "              " "
4  ( 1 )  " "              "*"                 " "              " "
5  ( 1 )  " "              "*"                 " "              " "
6  ( 1 )  " "              "*"                 " "              " "
7  ( 1 )  " "              "*"                 " "              " "
8  ( 1 )  " "              "*"                 " "              " "
9  ( 1 )  " "              "*"                 " "              " "
10  ( 1 ) " "              "*"                 " "              " "
11  ( 1 ) " "              "*"                 "*"              " "
```

Figure 4. The result of the selection

After it, using summary function on your Stepwise model you can extract the result of the selection and see which features to choose, as it possible to see on the Figure 4.

A data.frame: 18341 × 12

	pdays	cons.conf.idx	nr.employed	job_blue-collar	month_jul	month_jun	month_mar	month_may	day_of_week_mon	poutcome_nonexistent	poutcome_success	subscribe
	<int>	<dbl>	<dbl>	<int>	<int>	<int>	<int>	<int>	<int>	<int>	<int>	<int>
1	999	-42.7	5228.1	0	1	0	0	0	0	1	0	0
2	999	-41.8	5228.1	0	0	1	0	0	1	1	0	0
3	999	-46.2	5099.1	0	0	0	0	1	1	1	0	0
4	999	-47.1	5099.1	0	0	0	0	0	0	1	0	0
5	999	-36.4	5191.0	0	0	0	0	1	0	1	0	0
6	999	-36.4	5191.0	0	0	0	0	1	0	1	0	0
8	999	-36.4	5191.0	1	0	0	0	1	0	1	0	0
9	999	-46.2	5099.1	1	0	0	0	1	0	0	0	0

Figure 5. The final dataset

On the Figure 5 you can see the final dataset. It is possible to notice that the final number of columns has changed from 64 to 12.

Model Application

Machine learning is a form of artificial intelligence (AI) that teaches computers to think like humans: learn and improve based on past experience. It works by examining data and identifying patterns and requires minimal human intervention. Almost any task that can be accomplished with a pattern or set of data-driven rules can be automated using machine learning. This allows companies to transform processes that were previously only available to humans. One of the types of these kind of processes are classification problems.

Classification is a central topic in machine learning, concerned with teaching machines how to group data according to certain criteria. Classification is the process by which computers group data based on predetermined characteristics. Choosing the right classification algorithm is very important. The

algorithm that performs the classification is called a classifier. The classifier algorithm must be fast, accurate, and sometimes minimize the amount of training data needed. As a general rule, the more parameters a dataset has, the larger the training set for the algorithm should be. Different classification algorithms basically have different ways of learning patterns from examples. More formally, classification algorithms map observation *v* to concept *w*.

There are many different types of classification algorithms for modeling predictive classification modeling problems. There is no good theory about how to map algorithms to problem types; instead, it is generally recommended that the practitioner use controlled experiments and find out which algorithm and algorithm configuration provides the best performance for a given classification task.

In this case, according to the requirements of the project, the task was to choose five different models, explain them and then compare in order to see the best performance.

First model that was applied is KNN. The principle behind KNN classifier, K-Nearest Neighbor, algorithm is to find K predefined number of training samples that are closest in the distance to a new point & predict a label for our new point using these samples. The most commonly used distance measure is Euclidean distance. The Euclidean distance is also known as simply distance. The usage of Euclidean distance measure is highly recommended when data is dense or continuous. Euclidean distance is the best proximity measure. The KNN classifier is also considered an instance-based learning/non-generalization algorithm. It stores training data records in a multidimensional space. For each new sample and specific value of K, it recalculates the Euclidean distances and predicts the target class. Thus, it does not create a generalized internal model.

In addition to the KNN model and other models that will be described later, a Cross-Validation, CV, was used. Data scientists rely on several reasons for using CV in the process of building machine learning models. One of the most important pillars of validating a learning model before it goes into production is making accurate predictions on unseen data. Unseen data is all data types that the model has never learned before. Ideally, test data should feed directly into the model in many test iterations. However, access to such data is limited or not yet available in the new environment. But the most important reason for us in this project is tuning the hyperparameters. Finding the best combination of model parameters is a common step to tune an algorithm to learn hidden patterns in a dataset. But doing this step on a simple learning-testing split is generally not recommended. The performance of the model is usually very sensitive to such parameters and tuning them based on a predefined partition of the dataset should be avoided. This can lead to overfitting the model and reducing its ability to generalize.

So, due to the CV and checking hyperparameters, we are going to see first the result of the normal KNN model and the result after hyperparameters tuning.

```
Aggregated Result: acc.test.mean=0.9122599,auc.test.mean=0.7181007,brier.test.mean=0.0737542
```

```
Resample Result
Task: data_final
Learner: classif.kknn
Aggr perf: acc.test.mean=0.9122599,auc.test.mean=0.7181007,brier.test.mean=0.0737542
Runtime: 4.32983
```

Figure 6. Normal KNN result

On the Figure 6 it is possible to see the result of the KNN model with the standard amount of 100 neighbors. This model was made with use of *mlr* library plus *makeLearner* and *makeClassifTask* functions from that exact package. In order to determine the result of our model, here and later on, we are going to use Accuracy, AUC and Brier. Accuracy simply means the amount of the correct predictions to the overall number of predictions. Usually, Accuracy above 90 is a good sign, since very often if it is lower, then in the corporate world it won't be accepted. AUC means area under the curve. ROC can be quantified using AUC. This is done to see how much area is covered by the ROC curve. If we get an ideal classifier, then the AUC will be 1.0. If the classifier is random in their guesses, then the AUC is 0.5. The Brier Score is a scoring metric that is used to check whether a predicted probability score is correct. This is very similar to standard error, but only applies to predictive probability estimates that range from 0 to 1.

```
[Tune] Result: k=200 : acc.test.mean=0.9120874,auc.test.mean=0.7152026,brier.test.mean=0.0733476
```

Figure 7. KNN result after hyperparameters

On the Figure 7 you can see the result after the hyperparameters tuning. From this vector of possible values:

c(1, 5, 10, 20, 50, 100, 200, 300)

The algorithm determined that the best number of neighbors is 200.

In terms of convenience, to not repeat the same things, for the next models only the explanation of the algorithm will be present in the report, since the result and the values of the parameters tuning is available in the project itself.

The second model that was applied was a random forest. Random forest is a supervised learning algorithm. The "forest" it builds is an ensemble of decision trees, usually trained using the "bag" method. The general idea of the bagging method is that the combination of learning models increases the overall result. I other words, Random Forest builds multiple decision trees and combines them together to get a more accurate and stable prediction. One of the big advantages of random forest is that it can be used for both classification and regression problems, which make up most modern machine learning systems.

This model also was applied thought the *mlr* library. The same way as with the previous model, the cross validation was applied. For the hyperparameters tuning the value of *ntree*, which basically represents a number of trees inside of the forest, was settled to 400 as the best result.

The next model that was used is AdaBoosting. The AdaBoost algorithm, short for Adaptive Boosting, is a boosting technique used as an ensemble method in machine learning. This is called adaptive gain because weights are reassigned to each instance and higher weights are assigned to misclassified instances. Boosting is used to reduce bias as well as variance for supervised learning. It works on the principle of consistent learner growth. With the exception of the first, each subsequent learner grows out of previously grown learner. Simply put, weak students become strong ones. The AdaBoost algorithm works on the same principle as boosting, with a few differences. In our case AdaBoosting was applied with *ada* library together with *mlr* library as before. For the hyperparameters tuning the parameter *size*, which represents the number of iterations, was chosen. After the tuning its optimal value was set to 70.

Fourth model that was implemented was a neural network from the *nnet* library. The neural network loosely mimics the neurons in the human brain, the interaction of which is "fired" as we think. In the digital world, a neuron is just a node containing some value representing an activation, which you can think of as a degree of truth about something. Each node has one or more connections to other nodes. Neurons are usually organized into "layers", with the first layer looking at the input image, the second layer looking at the output of the first layer, and so on until the last layer, which produces the output of the neural network. For the hyperparameters tuning the parameter *size*, which represents the number amount of the hidden layers, was chosen. After the tuning its optimal value was set to 10.

And the last model that was used was a Regularized Discriminant Analysis (RDA), which was implemented with use of *klaR* library. RDA is basically an extension of Linear Discriminant Analysis (LDA). The LDA representation is direct. It consists of statistical properties of your data calculated for each class. For one input variable (x), this is the mean and variance of the variable for each class. For multiple variables, these are the same properties calculated from the multivariate Gaussian, namely the means and the covariance matrix. These statistical properties are estimated based on your data and included in the LDA equation for prediction. RDA introduces regularization into variance estimation, softening the impact of different variables on LDA. For the hyperparameters tuning the parameter *fold*, which represents the number of samples to draw for cross-validation or bootstrap, was chosen. After the tuning its optimal value was set to 15.

References

An Introduction to Statistical Learning (2017)

What Is Machine Learning - ML - and Why Is It Important? | NetApp. (2022). Retrieved 25 March 2022, from https://www.netapp.com/artificial-intelligence/what-is-machine-learning/

R, S. (2022). Stepwise Regression Essentials in R - Articles - STHDA. Retrieved 25 March 2022, from http://www.sthda.com/english/articles/37-model-selection-essentials-in-r/154-stepwise-regression-essentials-in-

r/#:~:text=The%20stepwise%20regression%20%28or%20stepwise%20selection%29%20consists%20of,that%20is%20a%20model%20that%20lowers%20prediction%20error

Machine Learning in Banking - Opportunities, Risks, Use Cases. (2022). Retrieved 25 March 2022, from https://spd.group/machine-learning/machine-learning-in-banking/

Effects of outliers on regression model? (2022). Retrieved 26 March 2022, from https://discuss.analyticsvidhya.com/t/effects-of-outliers-on-regression-model/2403/2

5 Ways to Find Outliers in Your Data - Statistics By Jim. (2022). Retrieved 26 March 2022, from https://statisticsbyjim.com/basics/outliers/#:~:text=Using%20Z%2Dscores%20to%20Detect%20Outliers&text=A%20Z%2Dscore%20of%20zero,value%20that%20equals%20the%20mean.&text=The%20further%20away%20an%20observation's,3%20or%20further%20from%20zero

Importance of Feature Selection in Machine Learning | Aretove. (2022). Retrieved 27 March 2022, from https://www.aretove.com/importance-of-feature-selection-in-machine-learning

Classification Problems | Brilliant Math & Science Wiki. (2022). Retrieved 29 March 2022, from https://brilliant.org/wiki/classification/

Brownlee, J. (2022). 4 Types of Classification Tasks in Machine Learning. Retrieved 29 March 2022, from https://machinelearningmastery.com/types-of-classification-in-machine-learning/#:~:text=In%20machine%20learning%2C%20classification%20refers,one%20of%20the%20known%20characters

KNN R, K-Nearest Neighbor implementation in R using caret package. (2022). Retrieved 29 March 2022, from https://dataaspirant.com/knn-implementation-r-using-caret-package/#:~:text=As%20in%20our%20Knn%20implementation%20in%20R%20programming,some%20great%20packages%20to%20make%20our%20work%20easier

The Ultimate Guide to AdaBoost Algorithm | What is AdaBoost Algorithm?. (2022). Retrieved 30 March 2022, from https://www.mygreatlearning.com/blog/adaboost-algorithm/

AI Machine Learning Algorithms – How a Neural Network Works. (2022). Retrieved 30 March 2022, from https://developer.qualcomm.com/blog/ai-machine-learning-algorithms-how-neural-network-works

Brownlee, J. (2022). Linear Discriminant Analysis for Machine Learning. Retrieved 30 March 2022, from https://machinelearningmastery.com/linear-discriminant-analysis-for-machine-learning/