Mathematics in Machine Learning

Politecnico di Torino

Mudassar Hussain
Student id: s281654
s281654@studenti.polito.it

Umar Farooq
Student id: s292448
s292448@studenti.polito.it

Summer Session 2021

Abstract

The input is a set of Monte Carlo data, generated and approximately triggered and pre-processed for an imaging gamma-ray Cherenkov telescope. Such data belong to two classes, originating either from incident gamma rays or caused by hadronic showers. There is only a weak discrimination between signal (gamma) and background (hadrons), making the data an excellent proving ground for classification techniques.

Our work focus on classification of signal caught by Cherenkov telescope into gamma signal or hadron (background). Simple classification in this case was not a good option as you will see afterwards that the data is not distributed uniformly but it is biased towards gamma signal. So to address this issue and we did some Exploratory Data Analysis (EDA), tried to overcome biasness using SMOTE and then used Various classifiers in order to get as good results as possible.

Contents

Chapter 1

Introduction

The MAGIC telescopes are one of the three major IACTs (Imaging Atmospheric Cherenkov Telescopes) for observation of gamma rays in the TeV regime currently operative. MAGIC functions since 2003, and has published data from more than 60 sources, mostly blazars. MAGIC already provides astronomical `.fits` files with basic final scientific products such as spectral energy distributions, light curves and skymaps from published results. The MAGIC telescope, with its mirror surface of 236m2, is the largest and most advanced Cherenkov telescope in the world and the ground instrument with the lowest threshold and highest sensitivity in the cosmic -ray energy domain below about 200GeV.

With a reflector diameter of 17 meters each, the two MAGIC telescopes are the most sensitive Cherenkov telescopes in the world, especially in the energy range below 200 gigaelectronvolts (GeV). Their line of sight is directed at objects that emit gamma rays ranging from 30 GeV to 100 TeV (teraelectronvolts). This means that MAGIC can cover an enormous energy spectrum.

The twin telescopes are located 2,200 meters above sea level on the Canary Island of La Palma where the clear skies and lack of light pollution make for optimal observing conditions. The Max Planck Institute for Physics (MPP) leads the international collaboration of about 165 astrophysicists from 24 research institutions in eleven countries. Together, they are responsible for the construction, operation and maintenance of the telescopes. MAGIC allows astrophysicists to obtain first class data for gaining scientific insights into enigmatic objects and the most violent processes in the universe.

The MAGIC telescopes have been in operation since 2003 and 2009 re-

spectively. The MPP played a major role in the development and construction of their mechanical structure, imaging cameras and calibration system.

Since the outset, MAGIC has delivered many valuable scientific discoveries.

For technical reasons, the number of h events is underestimated. In the real data, the h class represents the majority of the events.

Chapter 2

Exploratory Data Analysis

The input is a subset of Monte Carlo data, generated and approximately triggered and pre-processed for an imaging gamma-ray Cherenkov telescope. Such data belong to two classes, originating either from incident gamma rays or caused by hadronic showers.The dataset contains 11 columns and 19,020 instances.

- **fLength**: major axis of ellipse [mm]
- **fWidth**: minor axis of ellipse [mm]
- **fSize** : 10-log of sum of content of all pixels [in phot]
- **fConc**: ratio of sum of two highest pixels over fSize [ratio]
- **fConc1**: ratio of highest pixel over fSize [ratio]
- **fAsym**: distance from highest pixel to center, projected onto major axis [mm]
- **fM3Long**: 3rd root of third moment along major axis [mm]
- **fM3Trans**: 3rd root of third moment along minor axis [mm]
- **fAlpha** : 3rd root of third moment along minor axis [mm]
- **fDist** : distance from origin to center of ellipse [mm]
- **class** : g,h: gamma (signal), hadron (background)

All of the features are numerical except the one we have to classify. During preliminary analysis no missing values were found.

	fLength	fWidth	fSize	fConc	fConc1	fAsym	fM3Long	fM3Trans	fAlpha	fDist
count	19020.000000	19020.000000	19020.000000	19020.000000	19020.000000	19020.000000	19020.000000	19020.000000	19020.000000	19020.000000
mean	53.250154	22.180966	2.825017	0.380327	0.214657	-4.331745	10.545545	0.249726	27.645707	193.818026
std	42.364855	18.346056	0.472599	0.182813	0.110511	59.206062	51.000118	20.827439	26.103621	74.731787
min	4.283500	0.000000	1.941300	0.013100	0.000300	-457.916100	-331.780000	-205.894700	0.000000	1.282600
25%	24.336000	11.863800	2.477100	0.235800	0.128475	-20.586550	-12.842775	-10.849375	5.547925	142.492250
50%	37.147700	17.139900	2.739600	0.354150	0.196500	4.013050	15.314100	0.666200	17.679500	191.851450
75%	70.122175	24.739475	3.101600	0.503700	0.285225	24.063700	35.837800	10.946425	45.883550	240.563825
max	334.177000	256.382000	5.323300	0.893000	0.675200	575.240700	238.321000	179.851000	90.000000	495.561000

Figure 2.1: Statistics of all the Numerical Features

The target class distribution is shown in the figure below.

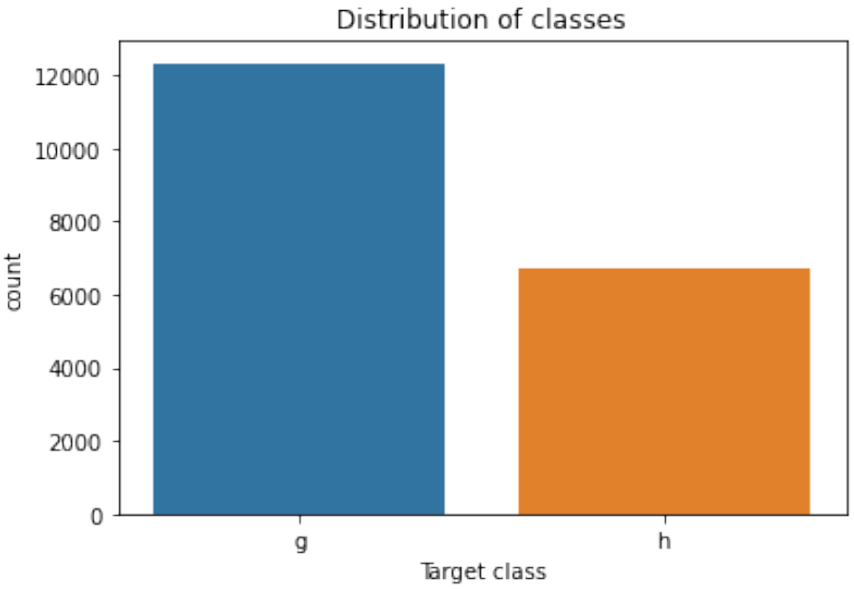


Figure 2.2: Target Class Distribution

We can clearly observe that the gamma signal class is almost double in count w.r.t hadron (background) class. And it is well-known that most of the algorithms are sensitive to unbalanced data, and the results out of the

algorithms will be biased towards majority class. We will see how we can overcome this issue later in preprocessing phase.

2.1 Box Plot

A boxplot is a standardized way of displaying the distribution of data based on a five number summary ("minimum", first quartile (Q1), median, third quartile (Q3), and "maximum"). It can tell us about outliers and their corresponding values. It can also tell us if the data is symmetrical, how tightly is the data grouped, and if and how the data is skewed. It would be more clear if we show the picture of boxplot.

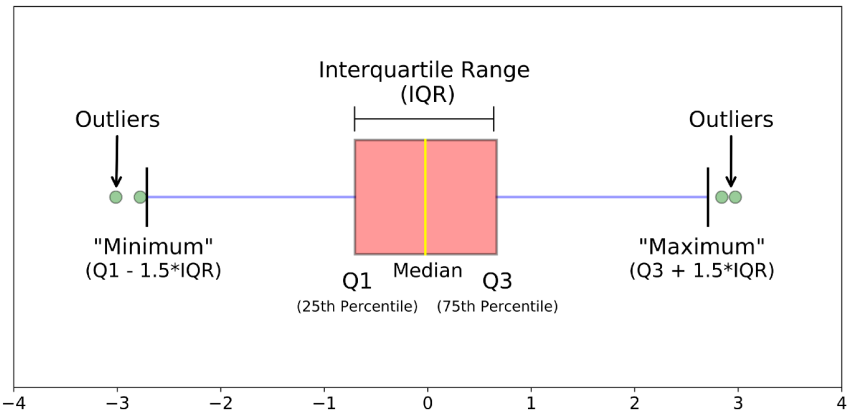


Figure 2.3: Box Plot

Distribution of all the features have been plotted in figure 2.4 using boxplots to extract useful insights.

2.1.1 Explanation for box plot

The plots reveal that almost every feature contains outliers, thus we will need to consider anomaly detection and removal during the preprocessing phase. An other observation is that features such as fAlpha, fLength, fWidth show slight variation in there distribution when plotted separately for each plot,

this observation gives us an idea of feature importance in a very early phase (i.e Data exploration).

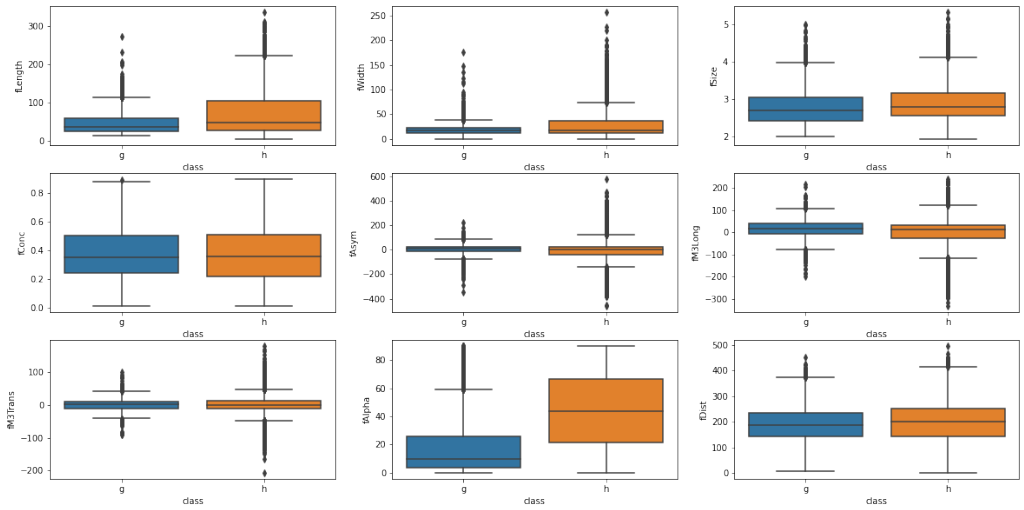


Figure 2.4: BoxPlots

2.2 Kernel Density Estimation

Kernel density estimation is a non-parametric model also know as KDE, it's a technique that lets you create a smooth curve given a set of data. KDE basically centers a kernel function at each data point and smooths it to get a destiny estimate. The motivation behind the creation of KDE was that Histograms are not smooth, they depend on the width of the bins and the endpoints of the bins, KDEs reduce the problem by providing smoother curves. This can be useful if you want to visualize just the "shape" of some data, as a kind of continuous replacement for the discrete histogram. Kernel Density Estimation smoothen the data by convolving each point with some 'kernel'. Each point is represented as the center for a kernel density function and the final curve is normalized convolution(sum) of all kernels at that point. It helps to visually observe that if the distribution of all the features are separate kernel density functions or not if yes, we might want to apply LDA on the data otherwise think of something else. The Kernel Density Estimation plots for the data have been in figure 2.5 :

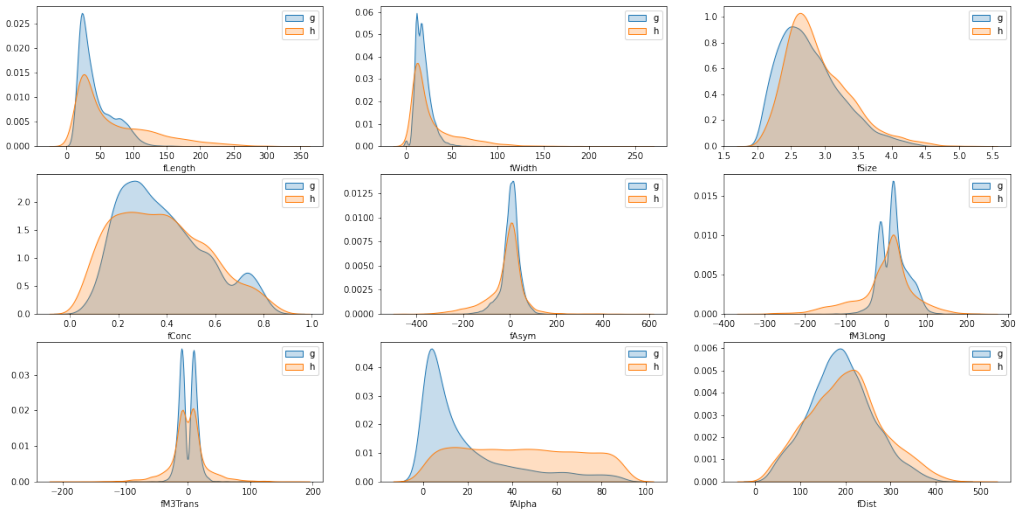


Figure 2.5: Feature Distributions

2.2.1 Explanation for KDE plots

Observing the KDE plots, reveals that the class wise distribution of features are overlapping. This type of distribution does not comply with algorithms such as Linear discriminant analysis where we assume that the inter class distributions are well separated. We will discuss more on that in the section of Dimensionality reduction.

2.3 PairPlot

Pair Plots are a really simple way to visualize relationships between each variable. It produces a matrix of relationships between each variable in the data for an instant examination of the data. A pair plot allows us to see both distribution of single variable and relationship between two variables. Pair plots are a great method to identify trends for follow-up analysis and, fortunately, are easily implemented in Python.The histogram on the diagonal allows us to see the distribution of a single variable while the scatter plots on the upper and lower triangles show the relationship (or lack thereof) between two variables. Here is the pairplot for our data.

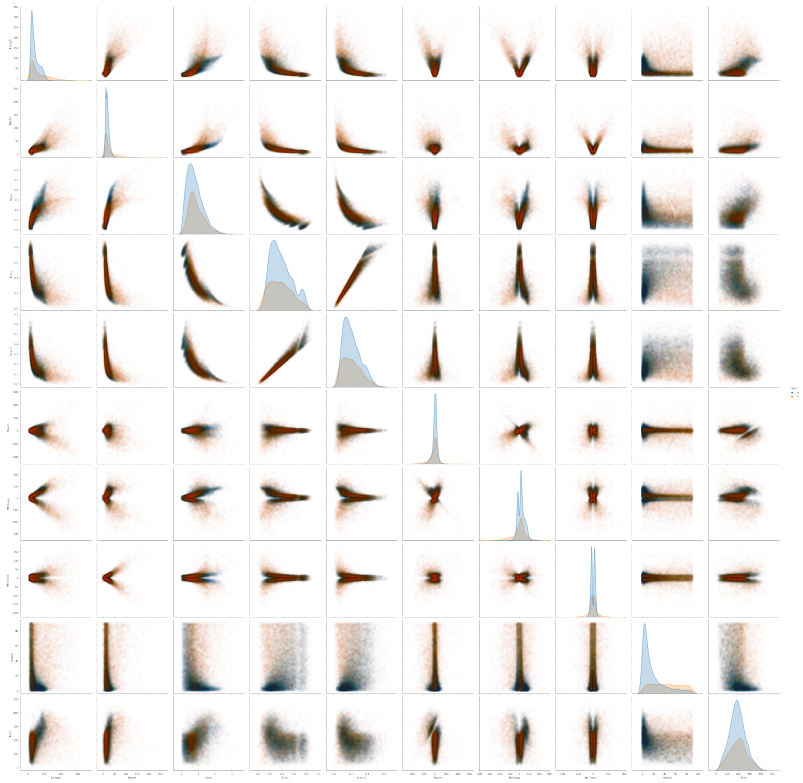


Figure 2.6: PairPlot

We used Kernel Density Estimation distribution plots at diagonals because the density plots on the diagonal make it easier to compare distributions between the class than stacked bars. Changing the transparency of the scatter plots increases readability because there is considerable overlap (known as overplotting) on these figures.

Pairs plots are a powerful tool to quickly explore distributions and relationships in a dataset. Seaborn provides a simple default method for making pair plots that can be customized and extended through the Pair Grid class. In a data analysis project, a major portion of the value often comes not in the flashy machine learning, but in the straightforward visualization of data. A pair plot provides us with a comprehensive first look at our data and is a unique tool in data analysis pipeline.

2.4 HeatMap

Heat Map Chart, or Heatmap is a two-dimensional visual representation of data, where values are encoded in colors, delivering a convenient, insightful view of information. Heat map for the dataset is given in figure 2.7. The map reveals that fconc and fconc1 are almost overlapping feature with a correlation of 0.98. Thus we can drop one of them without loss of information.Moreover, first three feature have a correlation greater then 0.7, we might want a compact representation by mapping these feature on some other dimension(s).More on this in the section of PCA.

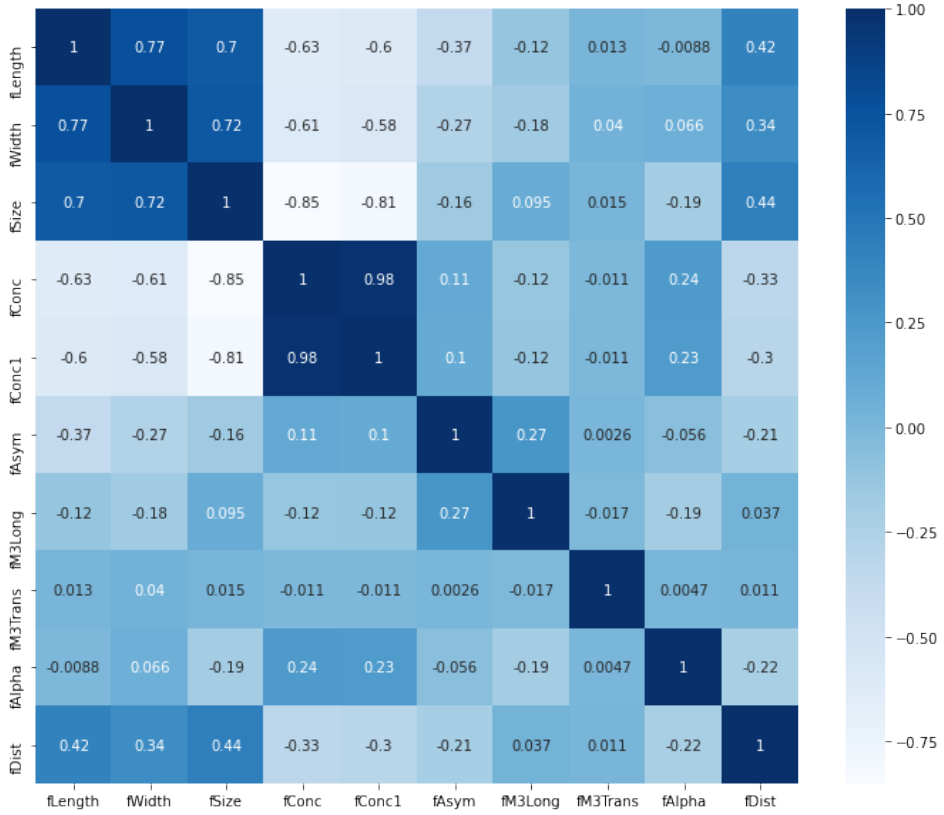


Figure 2.7: HeatMap

Chapter 3

Data Pre-Processing

3.1 Anomaly detection

3.1.1 Isolation Forest

Isolation forest is a machine learning algorithm for anomaly detection.It's an unsupervised learning algorithm based on the Decision Tree Classifier. It isolates the outliers by randomly selecting a feature from the given set of features and then randomly selecting a split value between the max and min values of that feature. In principle, outliers are less frequent than regular observations and are different from them in terms of values (they lie further away from the regular observations in the feature space). That is why by using such random partitioning they should be identified closer to the root of the tree (shorter average path length, i.e., the number of edges an observation must pass in the tree going from the root to the terminal node), with fewer splits necessary.

The idea of identifying a normal vs. abnormal observation can be observed in Figure 3.1 from [1]. A normal point (on the left) requires more partitions to be identified than an abnormal point (right).

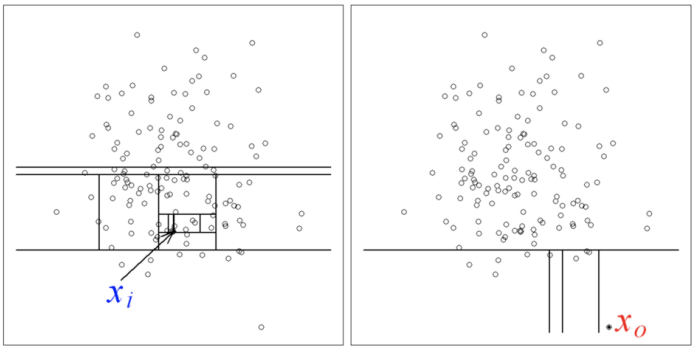


Figure 3.1: Identifying normal vs. abnormal observations

As with other outlier detection methods, an anomaly score is required for decision making. In the case of Isolation Forest, it is defined as:

$$s(x, n) = 2^{\frac{-E(h(x))}{c(n)}}$$

where h(x) is the path length of observation x, c(n) is the average path length of unsuccessful search in a Binary Search Tree and n is the number of external nodes.

Each observation is given an anomaly score and the following decision can be made on its basis:

- A score close to 1 indicates anomalies
- Score much smaller than 0.5 indicates normal observations
- If all scores are close to 0.5 then the entire sample does not seem to have clearly distinct anomalies

We used isolation forest for elimination of outliers and we got 1902 outliers eliminated from our data set and we remained with 17118 instances, as isolation forest eliminates entire row from dataset.

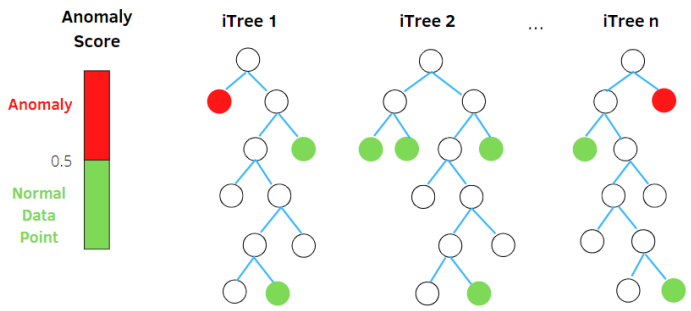


Figure 3.2: Isolation Forest

3.2 Principal Component Analysis

In real world data analysis tasks we analyze complex data i.e. multi dimensional data. We plot the data and find various patterns in it or use it to train some machine learning models. One way to think about dimensions is that suppose we have a data point x , if we consider this data point as a physical object then dimensions are merely a basis of view, like where is the data located when it is observed from horizontal axis or vertical axis. As the dimensions of data increases, the difficulty to visualize it and perform computations on it also increases. So, how to reduce the dimensions of a data-

- Only keep the most important dimensions

PCA finds a new set of dimensions (or a set of basis of views) such that all the dimensions are orthogonal (and hence linearly independent) and ranked according to the variance of data along them. It means more important principleaxis occurs first. (more important = more variance/more spread out data).

How does PCA work -

- Compute the vector μ containing mean of all the features in the dataset.
- Compute the centered data Matrix: $B = X - \mu$

- Calculate the covariance matrix of data: $S = \frac{1}{n}BB^T$
  Where S has a property of being symmetric and hence can be diagonalized. $S = PDP^T$

- Calculate the eigenvalues and eigenvectors over covariance matrix. Where we require that: $\lambda_1 \geq \lambda_2 ..... \geq \lambda_n \geq 0$

- Choose the principal components according to given threshold of explained varience (e.g 0.9) or number of components to be kept.

3.2.1 Covariance

It is a measure of the extent to which corresponding elements from two sets of ordered data move in the same direction.

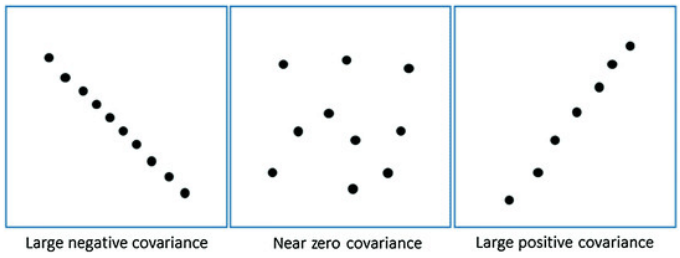


Figure 3.3: Covariance

Positive covariance means X and Y are positively related i.e. as X increases Y also increases. Negative covariance depicts the exact opposite relation. However zero covariance means X and Y are not related.

3.2.2 Eigenvectors and Eigenvalues

The eigenvectors and eigenvalues of a covariance (or correlation) matrix represent the "core" of a PCA: The eigenvectors (principal components) determine the directions of the new feature space, and the eigenvalues determine their magnitude. In other words, the eigenvalues explain the variance of the data along the new feature axes.

3.2.3 Dimensionality Reduction

PCA is an unsupervised dimensionality reduction technique that focuses on capturing most of the variance in a datset. Yet Linear discriminant analysis is another interesting tehcnique which is based on supervised learning. However, LDA assumes that the class wise distribution is well separated for each feature and this is not the case in our dataset. We already have less than 10 features but since the heatmap shows strong relation between fsize, fwidth and flength, we expect PCA to give us a more compact representation of the data.Explained varience and commmulative varience has been shown in figure 3.4.

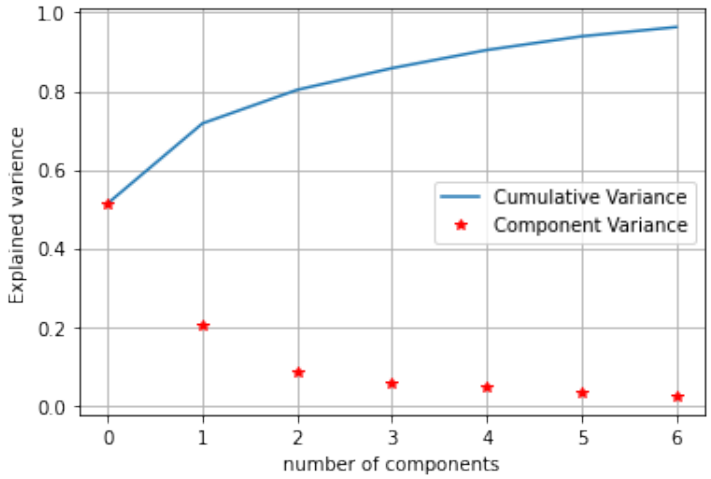


Figure 3.4: Principle Component Analysis

3.3 Cross Validation

Cross validation is a technique for assessing how the statistical analysis generalises to an independent data set.It is a technique for evaluating machine learning models by training several models on subsets of the available input data and evaluating them on the complementary subset of the data. Using cross-validation, there are high chances that we can detect over-fitting with ease.

3.3.1 K-Fold

Initially, the entire training data set is broken up in k equal parts. The first part is kept as the hold out (testing) set and the remaining k-1 parts are used to train the model. Then the trained model is then tested on the holdout set. The above process is repeated k times, in each case we keep on changing the holdout set. Thus, every data point get an equal opportunity to be included in the test set. Usually, k is equal to 3 or 5. It can be extended even to higher values like 10 or 15 but it becomes extremely computationally expensive and time-consuming.

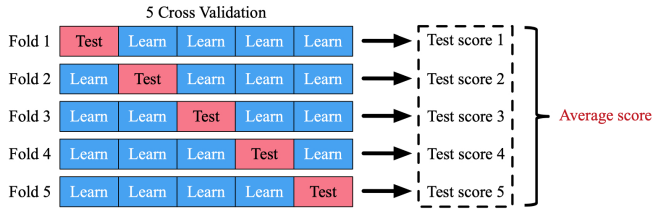


Figure 3.5: Crossvalidation with k=5

3.3.2 Sampling Bias

Sampling bias means that the samples of a stochastic variable that are collected to determine its distribution are selected incorrectly and do not represent the true distribution because of non-random reasons.Sampling bias often arises because certain values of the variable are systematically under-represented or over-represented with respect to the true distribution of the variable.

3.3.3 Stratified Sampling

To avoid sampling bias we have to consider stratified sampling which means that the sample from any specific class have to be of the same proportion in each split.Stratified sampling is different from simple random sampling, which involves the random selection of data from the entire population so that each possible sample is equally likely to occur. Stratified sampling ensures each subgroup within the population receives proper representation within the sample.As a result, stratified random sampling provides better coverage of the population.

3.4 Over sampling vs Under sampling

We often come across datasets where class distribution is not balanced. This results in the classifiers being biased towards the majority class. This situation can be dealt with either by under sampling the majority class or by over sampling the minority class. Under smapling will result in reduction of dataset instances which is not recommended. On the other hand we could oversmaple the minority class.There are several ways to do that. A very popular technique is known as SMOTE(Synthetic Minority Oversampling Technique).

Figure 3.6: Undersampling vs Oversampling

3.4.1 SMOTE

In this technique, new data points of the minority class are generated by taking a combination of existing points. We will use this technique during the training of all the classifiers. Figure shows working of this technique:

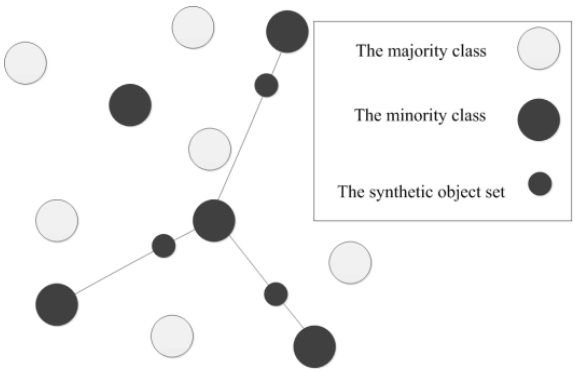


Figure 3.7: SMOTE

Although over smapling is very useful technique but has to be used in the pipeline after the cross validation splits, otherwise we may end up over fitting the model. This often occurs in case of bootstrap sampling where same instances are present both in the training and validation set due to over sampling. In any case the test and validation set should remain un touched. Figure 3.8 vissualy explains this step in the data analysis pipeline.

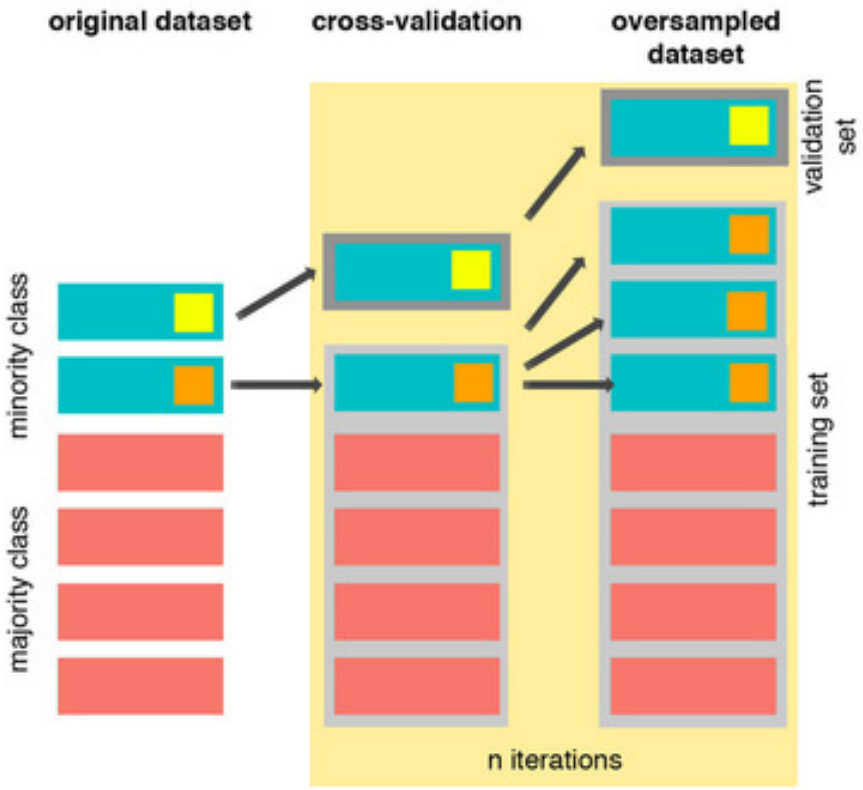


Figure 3.8: Sampling Cross Validation

Chapter 4

Model Selction

4.1 Decision Tree

Decision tree is a simple machine learning algorithm. The objective is to learn basic decision rules from data characteristics to construct a model that predicts the value of a target variable.

4.1.1 Advantages

- The model is highly interpretable. More over this technique does not require preprocessing such as data normalization, label encoding etc. However missing values still need to be handled.
- The cost of using the tree (i.e., predicting data) is logarithmic in the number of data points used to train the tree.

4.1.2 Disadvantages

- Decision-tree learners can create over-complex trees that do not generalise the data well thus creating the problem of overfitting.
- Decision tree learners create biased trees if some classes dominate. It is therefore recommended to balance the dataset prior to fitting with the decision tree.

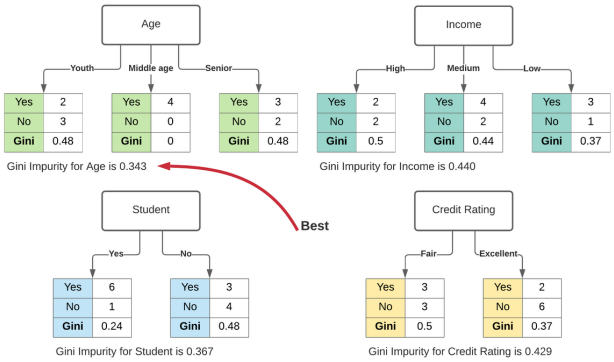
- Decision-tree learning algorithms are based on heuristics such as the greedy method where locally optimal decisions are made at each node. Such algorithms cannot guarantee to return the globally optimal decision tree.

4.1.3 Splitting Criteria

At each step, choice of the best predictor to split is made according to some criteria. These measures represent the purity of a split. The most common ones are:

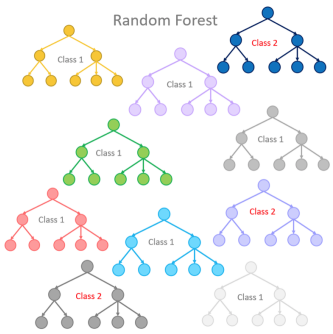
- Gini index: $1 - \sum_{i=1}^{n} (P_i)^2$
- Cross Entropy: $-\sum_{i=1}^{n} P_i log_2(P_i)$

Where Pi denotes the probability of an element being classified for a distinct class. Here is a visual example of gini.



4.2 Random Forest

Random forest classifier belong to the ensemble class. Several decision trees are trained, each of them being trained on a different set of bootstrapped training data. At each split, only m predictors participate (usually $\sqrt{p}$). This splitting results in uncorrelated trees and hence obtaining a more robust model. Final decision is made by majority voting of the participating trees.



4.2.1 Bootstrap aggregation

The Random Forest algorithm uses Bootstrap aggregating, also called bagging, as its ensembling method. It gains accuracy and combats overfitting by not only averaging the models but also trying to create models that are as uncorrelated as possible by giving them different training data sets. It creates the data-sets using sampling with replacement a straightforward but sufficient data sampling technique. Sampling with replacement means that some data-points can be picked multiple times.To further decrease the correlation between individual trees, each decision tree is trained on different randomly selected features. The number of features used for each individual tree is a hyperparameter, often called *max_features or n_features*.

So each decision tree in a random forest is not only trained on a different data-set (thanks to bagging) but also on different features/columns. After the individual decision trees are trained, they are combined together. For classification, max voting is used. That means the class, which most trees have as the output, will be the Random Forest's output. For regression, the outputs of the Decision Trees are averaged.

4.2.2 Hyper parameters

Random forests have several hyper parameters and it is important to understand their role in order to tune the classification model.

- **Number of Estimators:** The number of trees in the forest.
- **Split criteria:** The function to measure the quality of a split (Gini impurity or Entropy for classification and Mean-Squared-Error (MSE) or Mean-Absolute-Error (MAE) for regression).

- **Maximum depth:** The maximum depth of a tree.
- **Minimum samples per split:** The minimum number of samples required to split an internal node.
- **Minimum samples per leaf:** The minimum number of samples required to be at a leaf node.
- **Maximum number of features:** The number of features to consider when looking for the best split.

4.3 Logistic Regression

Logistic Regression is a Supervised statistical technique to find the probability of dependent variable(Classes present in the variable). Logistic regression uses functions called the logit functions,that helps derive a relationship between the dependent variable and independent variables by predicting the probabilities or chances of occurrence. The logistic functions (also known as the sigmoid functions) convert the probabilities into binary values which could be further used for predictions. The question may arise that why do we call it regression but using it as classifier? well the answer is simple, Logistic regression is a generalized linear regression model using the same basic formula of linear regression but it is regressing for the probability of a categorical outcome. The Logistic Regression instead for fitting the best fit line as in linear regression, condenses the output of the linear function between 0 and 1.

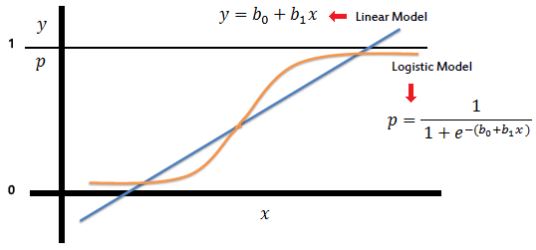


Figure 4.1: Logistic Regression

The function p is the logistic function, also known as the sigmoid function. In the formula of the logistic model, when $b_0 + b_1 x == 0$, then the p will be 0.5, similarly,$b_0 + b_1 x > 0$, then the p will be going towards 1 and $b_0 + b_1 X < 0$, then the p will be going towards 0.

As we can observe, the range of the sigmoid function is only between 0 and 1. This allows us to express data points as probabilities! More precisely, we can now map each (x, y) pair from a set of points into the graph of a sigmoid function and use it to generate probabilities as predictions.

4.3.1 Decision Boundaries

We expect our classifier to give us a set of outputs or classes based on probability when we pass the inputs through a prediction function and returns a probability score between 0 and 1. We basically decide with a threshold value above which we classify values into Class 1 and of the value goes below the threshold then we classify it in Class 2.

4.4 Support Vector Machine(SVM)

A Support Vector Machine (SVM) is a discriminative classifier formally defined by a separating hyperplane. In other words, given labeled training data (supervised learning), the algorithm outputs an optimal hyperplane which categorizes new examples. In two dimentional space this hyperplane is a line dividing a plane in two parts where in each class lay in either side. SVM or Support Vector Machine is a linear model for classification and regression problems. It can solve linear and non-linear problems and work well for many practical problems. In this algorithm, we plot each data item as a point in n-dimensional space (where n is number of features you have) with the value of each feature being the value of a particular coordinate. Then, we perform classification by finding the optimal hyper-plane that differentiate the two classes very well. lets take an example, suppose that we have 2-D data, classes are linearly separable and we want to separate them, there may be many solutions to the problem, as shown in the figure.

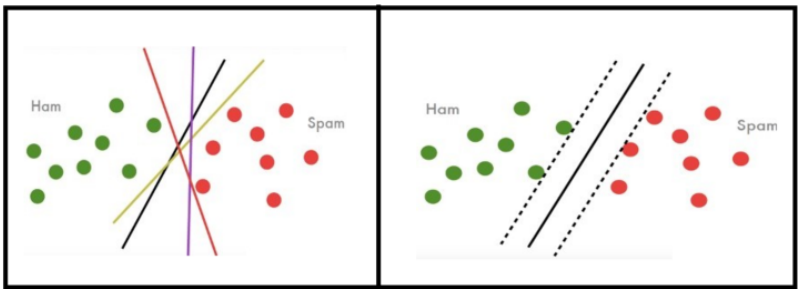


Figure 4.2: SVM separating 2-D data linearly

Now, here we have many solutions which one to choose? If we were using perceptron all of the solutions were optimal solutions as it was separation the classes but in the case of SVM it is different. The operation of the SVM algorithm is based on finding the hyperplane that gives the largest minimum distance to the training examples. Twice, this distance receives the important name of margin within SVM's theory. Therefore, the optimal separating hyperplane maximizes the margin of the training data. as shown in the second image where black line is separating Ham and Spam. The distance between the two dotted black lines is called the Maximum margin.

There can be a case where the data is not linearly separable for example.

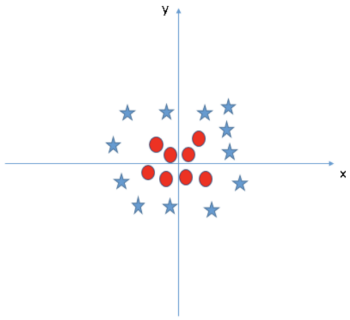


Figure 4.3: 2-D data linearly not separable

As you can see there does not exist a line that can separate the two classes. But in SVM, we can use some kernel functions that help us converting not linearly separable data into linearly separable. for example this data can be converted into linearly separable by using this simple kernel function

$$z = x^2 + y^2$$

Using the given kernel we were able linearly separate data as shown below.

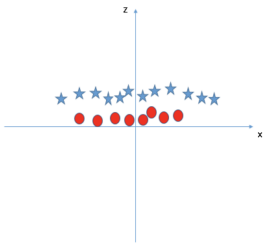


Figure 4.4: 2-D data linearly separable after using kernel function

Having real data, we might have some noise so we allow SVM to make a certain number of mistakes and keep margin as wide as possible so that other points can still be classified correctly. This can be done simply by modifying the objective of SVM. As shown in the figure below.

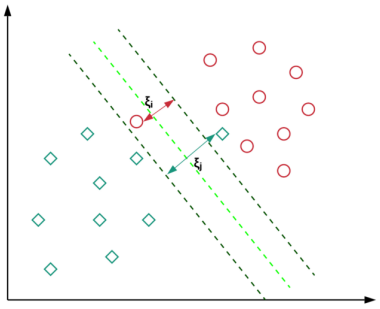


Figure 4.5: Linearly inseparable but SVM with tolerance

Observing all these characteristics we applied SVM on our data set and the results are given in the results section.

4.5 K-Nearest Neighbour

We often judge people by their vicinity to the group of people they live with. People who belong to a particular group are usually considered similar based on the characteristics they possess. This is the simple principle on which the KNN algorithm works — "Birds of the same feather flock together."
K-nearest neighbors is a supervised machine learning algorithm often used in classification problems. It works on the simple assumption that "The apple does not fall far from the tree" meaning similar things are always in close proximity. This algorithm works by classifying the data points based on how the neighbors are classified. Any new case is classified based on a similarity measure of all the available cases. Technically, the algorithm classifies an unknown item by looking at k of its already -classified, nearest neighbor items by finding out majority votes from nearest neighbors that have similar attributes as those used to map the items.

- **Lazy Learning Algorithm**: It is a lazy learner because it does not have a training phase but rather memorizes the training dataset. All computations are delayed until classification.
- **Case-Based Learning Algorithm**: The algorithm uses raw training instances from the problem domain to make predictions and is often referred to as an instance based or case-based learning algorithm. Case-based learning implies that KNN does not explicitly learn a model. Rather it memorizes the training instances/cases which are then used as "knowledge" for the prediction phase.
- **Non parametric**: It does not make an assumption about the underlying data distribution pattern.

4.5.1 Selection of K in KNN

K in KNN is the number of nearest neighbors considered for assigning a label to the current point. K is an extremely important parameter and choosing the value of K is the most critical problem when working with the KNN algorithm. The process of choosing the right value of K is referred to as parameter tuning and is of great significance in achieving better accuracy. If the value of K is too small then there is a probability of overfitting the model

and if it is too large then the algorithm becomes computationally expensive. Selecting the value of K depends on individual cases and sometimes the best method of choosing K is to run through different values of K and verify the outcomes. Using cross-validation, the KNN algorithm can be tested for different values of K and the value of K that results in good accuracy can be considered as an optimal value for K.

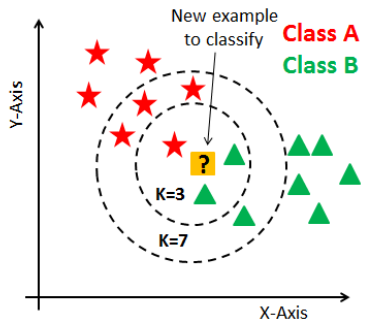


Figure 4.6: What K to choose?

As you can verify from the above image, if we proceed with K=3, then we predict that test input belongs to class B, and if we continue with K=7, then we predict that test input belongs to class A.

4.5.2 Determining a point as neighbor

We said that we classify a point in data set to its nearest neighbors but what does near mean? Well, in this case it means how much similar/close is the data point to the training samples in distance. Distance can be calculated using

- Euclidean distance
- Manhattan distance
- Hamming Distance

- Minkowski Distance

The idea to use distance measure is to find the distance (similarity) between new sample and training set and then finds the k-closest samples to new data point, all these k data points that are closest to the given point are known as k-nearest neighbors of that point.

4.5.3 Working

Working principles of KNN classifier is given in the points below.

- Choose a value for K. K should be an odd number.
- Find the distance of the new point to each of the training data.
- Find the K nearest neighbors to the new data point.
- Count the number of data points in each category among the k neighbors. New data point will belong to class that has the most neighbors.

KNN algorithm is a good choice if you have a small dataset and the data is noise free and labeled. When the data set is small, the classifier completes execution in shorter time duration. If your dataset is large, then KNN, without any hacks, is of no use. And in our case the data set was not very large and we also removed outlier using isolation forest hence we could easily afford to use KNN.

Here we have k=5 and 4 of the neighbors of the yellow data point belong to M class where 1 of them belongs to L class so we will assign M class to the new data point.

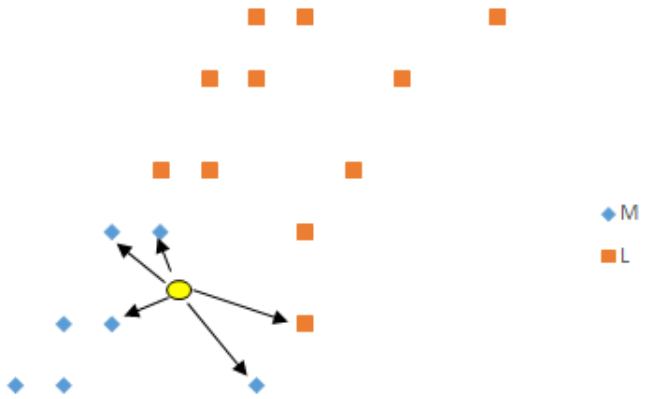


Figure 4.7: K-Nearest Neighbours classifier

Chapter 5

Evaluation metrics

Metric evaluations are performed in order to understand how well a classifier has performed. Mainly we use the following evaluation techniques:

5.1 Some Notations

Assuming that we have two classes A,B

- TP (True Positive) : when point belongs to class A and was predicted as class A. We say it is True Positive for class A.
- FP (False Positive) : when point doesn't belong to class A but was predicted as class A. We say it is False Positive for class A.
- TN (True Negative) : when point doesn't belong to class A and was predicted as class B. We say it is True Negative for class A.
- FN (False Negative) : when point belongs to class A but was predicted as class B. We say it is False Negative for class A.
- TPR (True Positive Rate) : Ability of classification model to identify all relevant instances.So when it is actually A,how many times it was predicted A.
- FRP (False Positive Rate) : When it is actually A ,how often it is predicted B.

5.2 Accuracy

Accuracy is the ratio of correctly predicted label with respect to the total number of instances in the test set.

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

The evaluation can lead to misleading conclusions. This may happen when one or more classes in the dataset are under represented.For example in a binary classification with classes A and B where A is present 90 percent. If a classifier always predicts A, its accuracy will still be 0.9.

5.3 Recall

Sometimes we are interested in the ratio of predicted true positives over total positive cases. This measure is particularly useful for health and medical services. For example one might be interesting in knowing how good the classifier performs in diagnosis of a disease among all the real cases of disease.

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

5.4 Precision

yet another perspective of analyzing the results of a classifier is the ratio between predicted true positive cases over total cases that were predicted positive.

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

5.5 F1-score

The F1 score can be interpreted as a weighted average of the precision and recall, where an F1 score reaches its best value at 1 and worst score at 0. The relative contribution of precision and recall to the F1 score are equal.

$$Precision = 2\frac{Precision*Recall}{Precision+Recall}$$

Chapter 6

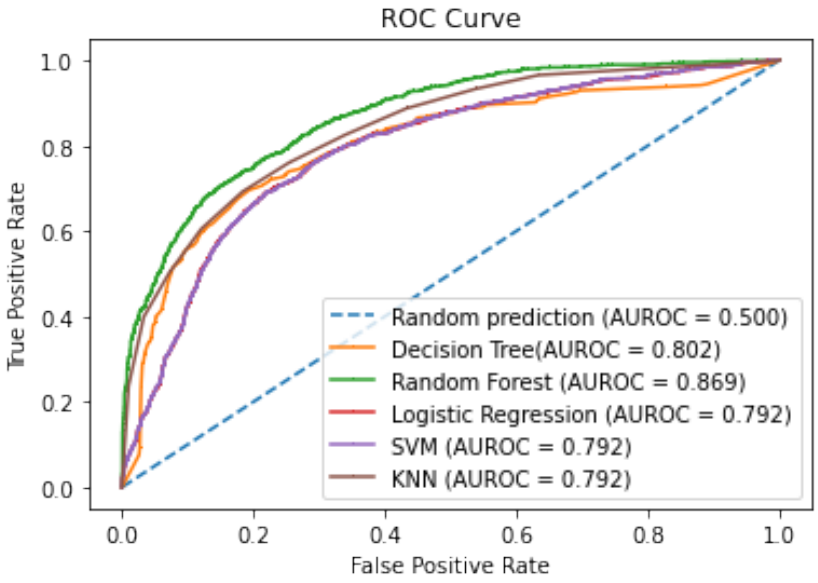
Model Evaluation

The table describes the results of metric evaluation on the classifiers.

Metric	Decision Tree	Random forest	Logistic Regression	SVM	K-NN
Accuracy	0.75	0.79	0.75	0.75	0.78
Recall	0.56	0.61	0.55	0.55	0.60
Precision	0.72	0.74	0.69	0.69	0.69
F1-score	0.63	0.67	0.61	0.61	0.64

6.1 Roc Curve

ROC curve is one the important evaluating metrics that should be used to check the performance of an classification model.It is also called relative operating characteristic curve, because it is a comparison of two main characteristics (TPR and FPR). A receiver operating characteristic curve, or ROC curve, is a graphical plot that illustrates the diagnostic ability of a binary classifier system as its discrimination threshold is varied.The ROC curve is created by plotting the true positive rate (TPR) against the false positive rate (FPR) at various threshold settings. Figure compares the evaluation of classifiers on the ROC curve. Random forest performs better with most AUC. The AUC is the area under the ROC curve. This score gives us a good idea of how well the model performances.



6.2 Confusion Matrix

Confusion Matrix is a performance measurement for machine learning classification problem where output can be two or more classes. It is a table with 4 different combinations of predicted and actual values. It is extremely useful for measuring Recall, Precision, Specificity, Accuracy, and most importantly AUC-ROC curves.

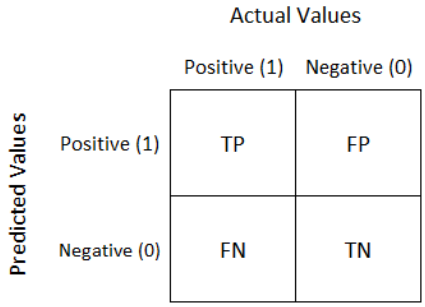


Figure 6.1: Confusion Matrix

6.3 Decision Tree

The confusion matrix for decision tree is given below.

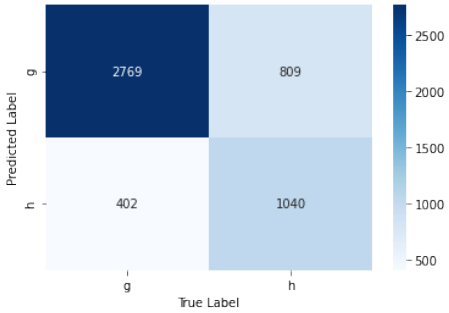


Figure 6.2: Decision Tree

6.4 Random Forest

The confusion matrix for Random Forest is given below.

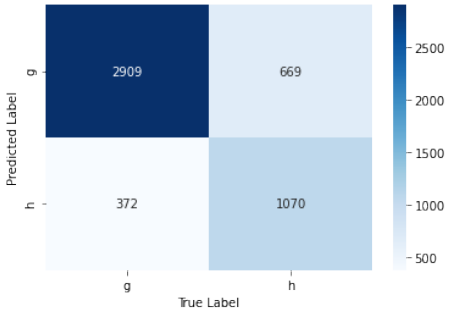


Figure 6.3: Random Forest

6.5 Logistic Regression

The confusion matrix for Logistic Regression is given below.

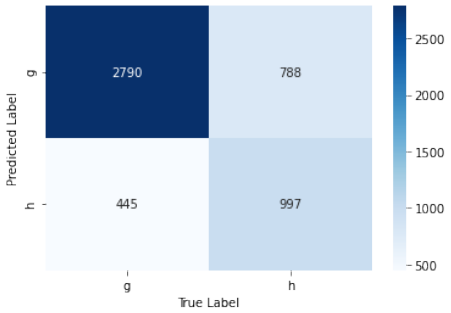


Figure 6.4: Logistic Regression

6.6 Support Vector Machine

The confusion matrix for Support Vector Machine is given below.

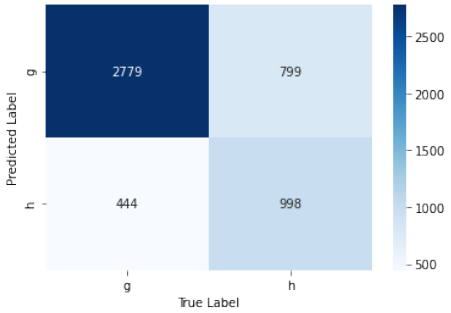


Figure 6.5: SVM

6.7 K-Nearest Neighbors

The confusion matrix for K-Nearest Neighbors is given below.

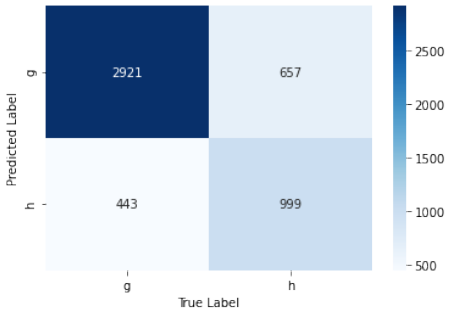


Figure 6.6: K-NN

Bibliography

[1] Liu, F. T., Ting, K. M., Zhou, Z. H. (2008, December). Isolation forest. In Data Mining, 2008. ICDM'08. Eighth IEEE International Conference on (pp. 413–422). IEEE.