Nicusor-Daniel Vlasin

R00124330

Machine Learning

Assignment 2

Table of Contents

[Introduction 2](#_Toc498980173)

[Related Research 3](#_Toc498980174)

[Algorithm/Model Detail 4](#_Toc498980175)

[Principal Component Analysis (PCA) 4](#_Toc498980176)

[Logistic Regression 4](#_Toc498980177)

[Gaussian Naive Bayes 6](#_Toc498980178)

[Support Vector Machine(SVC) 7](#_Toc498980179)

[Advantages of support vector machines 7](#_Toc498980180)

[Disadvantages of support vector machines 7](#_Toc498980181)

[Benefits over GridSearchCV 8](#_Toc498980182)

[Random Forest Classifier 9](#_Toc498980183)

[Decision tree model 10](#_Toc498980184)

[Empirical Evaluation 11](#_Toc498980185)

[Plotting boxplot 11](#_Toc498980186)

[Feature Selection 12](#_Toc498980187)

[Heatmap 13](#_Toc498980188)

[Interpretation of results 14](#_Toc498980189)

[Logic Regresion 14](#_Toc498980190)

[Naive Bayes 14](#_Toc498980191)

[Support Vector Machine(SVC) 14](#_Toc498980192)

[Decision tree model 14](#_Toc498980193)

# Introduction

Automatic text classification has always been an important application and research topic since the inception of digital documents. Today, text classification is a necessity due to the very large amount of text documents that we have to deal with daily.

In general, text classification includes topic based text classification and text genre-based classification. Topic-based text categorization classifies documents according to their topics Texts can also be written in many genres, for instance: scientific articles, news reports, movie reviews, and advertisements.

Genre is defined on the way a text was created, the way it was edited, the register of language it uses, and the kind of audience to whom it is addressed. Previous work on genre classification recognized that this task differs from topic-based categorization.

However, there are other scenarios, for examples one documents take part in one or more than two classes. In this project I will use the vector representation and classification. For vector representation I will use Sklearn python packages.

# Related Research

More formally, if D is a document of the entire set of documents D and {C1, C2, ,…., Cn} is the set of all the categories, then text classification assigns one category Cj to a document D. Machine Learning has been used to classify documents according to their topic.

In automatic text categorization, a machine learning model is given a set of examples of documents of different topics or genres and by uses these examples to build a model to classify those documents or piece of text to their class.

A screenshot of a cell phone

Description generated with very high confidence

The architecture of text categorization in the fig from above, the arrow with dashed line represents the data flow in the categorization process and the arrow with the solid line represents the data flow in the classifier construction process.

The task of constructing a classifier for documents does not differ a lot from other tasks of Machine Learning. The main issue is the representation of a document. One particularity of the text categorization problem is the vast number of feature (words and phrases) that can easily explode in order of ten of thousands. A way to solve this problem is to select a subset of features or transform the features into numbers. This method is called Vector Representation method which include of transforming the features into vector of numbers.

# Algorithm/Model Detail

## Principal Component Analysis (PCA)

Principal components analysis (PCA) is a standard linear technique for dimensionality reduction. Given a matrix X ∈ R n×l of l centered, n-dimensional observations, PCA performs an eigen decomposition of the covariance matrix Q := XXt.

This is a new method to perform nonlinear form of PCA by the use of integral operator kernel functions, one can efficiently compute principal components in high dimensional feature spaces. An example might be the spaces of all possible d-pixel products in images.

## Logistic Regression

In the multiclass case, the training algorithm uses the one-vs-rest (OvR) scheme if the ‘multi\_class’ option is set to ‘ovr’, and uses the cross- entropy loss if the ‘multi\_class’ option is set to ‘multinomial’.

Logistic regression is a statistical method for analysing a dataset in which there are one or more independent variables that determine an outcome. The outcome is measured with a dichotomous variable (in which there are only two possible outcomes).

In logistic regression, the dependent variable is binary or dichotomous, i.e. it only contains data coded as 1 (TRUE, success, eatable, etc.) or 0 (FALSE, failure, poison, etc.).

The goal of logistic regression is to find the best fitting model to describe the relationship between the dichotomous characteristic of interest and a set of independent (predictor or explanatory) variables.

This class implements regularized logistic regression using the ‘liblinear’ library, ‘newton-cg’, ‘sag’ and ‘lbfgs’ solvers. It can handle both dense and sparse input. Use C-ordered arrays or CSR matrices containing 64-bit floats for optimal performance; any other input format will be converted.

A screen shot of a clock

Description generated with high confidence

**C** implements uses a random number generator to select features when fitting the model.

**L1** and **L2** are regularization parameters. They’re used to avoid overfitting. Both L1 and L2 regularization prevents overfitting by shrinking (imposing a penalty) on the coefficients.

**L1** is the first moment norm |x1-x2| (|w| for regularization case) that is simply the absolute distance between two points whereL2 is second moment norm corresponding to Euclidean Distance that is |x1-x2|^2 (|w|^2 for regularization case).

In simple words, **L2 (Ridge)** shrinks all the coefficient by the same proportions but eliminates none, while **L1 (Lasso)** can shrink some coefficients to zero, performing variable selection. If all the features are correlated with the label, ridge outperforms lasso, as the coefficients are never zero in ridge. If only a subset of features is correlated with the label, lasso outperforms ridge as in lasso model some coefficient can be shrunken to zero.

A close up of a screen

Description generated with high confidence

The grid search provided by GridSearchCV exhaustively generates candidates from a grid of parameter values specified with the tuned\_parameter. The GridSearchCV instance implements the usual estimator API: when “fitting” it on a dataset all the possible combinations of parameter values are evaluated, and the best combination is retained.

Logistic Regression mode successfully classifies all the mushrooms with an accuracy of 100%, and established the best settings.

## Gaussian Naive Bayes

Naïve Bayes is simple, but power algorithm used for prediction modelling. In machine learning we are often interested in selecting the best hypothesis (h) given data (d). In a classification problem, our hypothesis (h) may be the class to assign for a new data instance (d). Bayes Theorem give us one of the easiest ways of selecting the most probable hypothesis and calculate the probability of that hypothesis give the prior knowledge.

**P(h|d) = (P(d|h) \* P(h)) / P(d)**

* **P(h|d)** is the probability of hypothesis h given the data d. This is called the posterior probability.
* **P(d|h)** is the probability of data d given that the hypothesis h was true.
* **P(h)** is the probability of hypothesis h being true (regardless of the data). This is called the prior probability of h.
* **P(d)** is the probability of the data (regardless of the hypothesis).

Naive Bayes is a classification algorithm for binary (two-class) and multi-class classification problems. Naive Bayes can be extended to real-valued attributes, most commonly by assuming a Gaussian distribution.

This extension of naive Bayes is called Gaussian Naive Bayes. Other functions can be used to estimate the distribution of the data, but the Gaussian (or Normal distribution) is the easiest to work with because you only need to estimate the mean and the standard deviation from your training data.

In this example a simple GaussianNB gave me the best result and wasn’t necessary to perform a grid search for it.

## Support Vector Machine(SVC)

In many supervised machines learning algorithms, labelling the data to train a model can be time consuming. Normally, the training set is chosen to be a random sampling of data. In the case of SVC, the model can actively choose the training data. This way its allowing the model an extra flexibility and will reduce the model need for large quantities of labelled data.

Support vector machines have strong theoretical foundations and excellent empirical successes. They have been applied to tasks such as handwritten digit recognition, object recognition, and text classification.

SVC is taking a training data {X1 ... Xn} that are vectors in some space then also, is taking their labels {Y1 ...Yn}. All vectors lying on one side ofthe hyperplane are labelled as −1, and all vectors lying on the other side are labelled as 1. The training instances that lie closest to the hyperplane are called support vectors.

### Advantages of support vector machines

* Effective in high dimensional spaces.
* Still effective in cases where number of dimensions is greater than the number of samples.
* Uses a subset of training points in the decision function (called support vectors), so it is also memory efficient.
* Versatile: different Kernel functions can be specified for the decision function. Common kernels are provided, but it is also possible to specify custom kernels.

### Disadvantages of support vector machines

* If the number of features is much greater than the number of samples, avoid over-fitting in choosing Kernel functions and regularization term is crucial.
* SVMs do not directly provide probability estimates, these are calculated using an expensive five-fold cross-validation

A screen shot of a computer

Description generated with high confidence

The **gamma** parameter defines how far the influence of a single training example reaches, with low values meaning ‘far’ and high values meaning ‘close’. The gamma parameters can be seen as the inverse of the radius of influence of samples selected by the model as support vectors.

The **C** parameter trades off misclassification of training examples against simplicity of the decision surface. A low C makes the decision surface smooth, while a high C aims at classifying all training examples correctly by giving the model freedom to select more samples as support vectors.

A screen shot of a person

Description generated with high confidenceA grid search provided by GridSearchCV will be the best solution in this case because the grid search exhaustively generates candidates from a grid of parameters values specified by the user.

The GridSearchCV instance implements the usual estimator API: when “fitting” it on a dataset all the possible combinations of parameter values are evaluated, and the best combination is retained. But in this situation due to computation to be too expensive and the lack of processing power in my laptop I am opting for a RandomizedSearchCV

RandomizedSearchCV is implementing a random set of settings (parameters) where each setting is a sample from a distribution over possible parameters values.

### Benefits over GridSearchCV

RandomizedSearchCV has two main benefits over an exhaustive search:

* A budget can be chosen independent of the number of parameters and possible values
* Adding parameters that do not influence the performance does not decrease efficiency.

A screen shot of a clock

Description generated with very high confidence

## Random Forest Classifier

A random forest is a meta estimator that fits a number of decision tree classifiers on various sub-samples of the dataset and use averaging to improve the predictive accuracy and control over-fitting. The sub-sample size is always the same as the original input sample size, but the samples are drawn with replacement if bootstrap=True.

The default values for the parameters controlling the size of the trees (e.g. max\_depth, min\_samples\_leaf, etc.) lead to fully grown and unpruned trees which can potentially be very large on some data sets. To reduce memory consumption, the complexity and size of the trees should be controlled by setting those parameter values.

The features are always randomly permuted at each split. Therefore, the best-found split may vary, even with the same training data.

In Laymen’s term,

Suppose training set is given as: [X1, X2, X3, X4] with corresponding labels as [L1, L2, L3, L4], random forest may create three decision trees taking input of subset for example,

[X1, X2, X3]

[X1, X2, X4]

[X2, X3, X4]

So finally, it predicts based on the majority of votes from each of the decision trees made. This works well because a single decision tree may be prone to a noise, but aggregate of many decision trees reduce the effect of noise giving more accurate results.

A number of tree features can be tuned to improve the performance of Random Forest model.

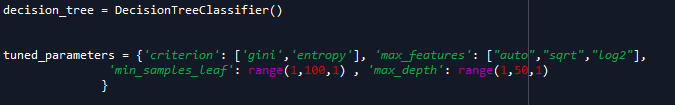
* max\_features: The maximum number of features Random Forest is allowed to try in individual tree.
  + Auto: This feature will simply take all the features which make sense in every tree.
  + Sqrt: This feature will take square root of the total number of features in each individual run.
  + Log2: It is another option which takes log to the base 2 of the features input.
* n\_estimators: This is the number of trees you want to build before taking the maximum voting or averages of predictions. Higher number of trees give you better performance but makes your code slower.
* min\_sample\_leaf: Leaf is the end node of a decision tree. A smaller leaf makes the model more prone to capturing noise in train data. Hence it is important to try different values to get good estimate.

## Decision tree model

Decision Trees) are a non-parametric supervised learning method used for classification and regression. The goal is to create a model that predicts the value of a target variable by learning simple decision rules inferred from the data features.

The DecisionTreeClassifier takes as input two arrays:

* An array X of size [n\_samples, n\_features] holding the training samples,
* An array Y of integer values, size [n\_samples], holding the class labels for the training samples.

**Criterion:** Decision trees use multiple algorithms to decide to split a node in two or more sub-nodes. Decision tree splits the nodes on all available variables and then selects the split which results in most homogeneous sub-nodes.

**max\_depth(Maximum depth of tree (vertical depth)):** Used to control over-fitting as higher depth will allow model to learn relations very specific to a particular sample.

**max\_features** and **min\_samples\_leaf** for those parameters I use the same as Random Forest classifier.

A close up of a screen

Description generated with high confidence

# Empirical Evaluation

Plotting boxplot

A screenshot of a cell phone

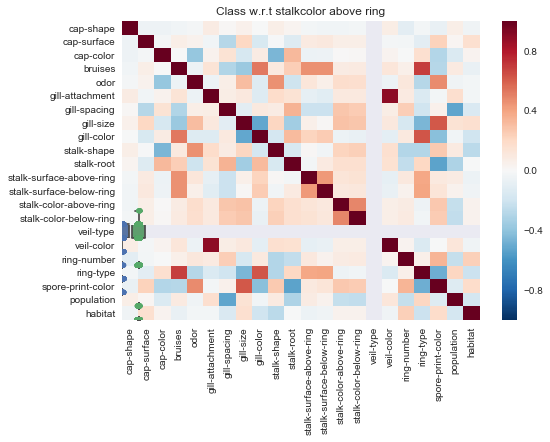
Description generated with very high confidence

## Feature Selection

A close up of a person

Description generated with high confidenceThis representation of the feature from the dataset shows that the first 3 feature are the most important because have the biggest value from all. For this feature selection i set the value of n\_feature = 1. In this situation the classification will be limited at only first 10 features from the dataset.

## Heatmap

One of the simplest approaches to feature selection is to examine the correlations between each of the individual features in your dataset. Features that display a very high correlation are candidates for removal.

There are many ways to plot this correlation between the features. In this example I use the corr() function available in a Panda data frame object to calculate the correlations and then use Seaborn to create a heatmap.

## Interpretation of results

### Logic Regresion

**Logic Regresion result: 1.0 %**

**Best parameters for this model are: {'C': 100, 'penalty': 'l2'}**

### Naive Bayes

**Number of mislabelled points from 1625 points: 111**

**Naive Bayes result: 0.846283402289 %**

### Support Vector Machine(SVC)

**Support Vector Machine best result: 1.0%**

**Best parameters for this model are: {'kernel': 'rbf', 'gamma': 0.1, 'C': 10}**

### Decision tree model

Unfortunately, I was unable to get the results and the best parameters due to the lack of processing power of my laptop.

# Conclusion and Future Work

In this project I applied 5 supervised machine learning models on the selected dataset. The strategy was to apply default model first with no tuning of the hyperparameter and then tuned them with different hyperparameter values.

The models used are as follows:

1. Principal Component Analysis
2. Logistic Regression
3. Gaussian Naive Bayes
4. Support Vector Machine
5. Random Forest Classifier
6. Decision trees

I was able to get the results from all of the module but the Decision Tree algorithm. Due to the extreme processing power required by the module my laptop was unable to compensate this power and the result was never displayed.

In the future I would like to work with some face recognize algorithms or some voice recognize models.

In automatic text classifica-

tion, a machine-learning algorithm is given a set of examples

of documents of different topics and uses these examples to

learn to distinguish documents. We consider the use of

machine-learning techniques to automatically categorize

documents according to genre.

In automatic text classifica-

tion, a machine-learning algorithm is given a set of examples

of documents of different topics and uses these examples to

learn to distinguish documents. We consider the use of

machine-learning techniques to automatically categorize

documents according to genre