Machine Learning and Pattern Recognition project:
Wine Quality classification task
Marco D'Almo (s301199)

With this report we are analyzing the results of classification tasks with different approaches, mainly implementing methods as seen in the Machine Learning and Pattern Recognition course at Politecnico di Torino.
The challenge is to classify perceived wine quality by humans by means of their physicochemical properties. Originally the wines were classified with a grade from 0 to 10, with 0 being extremely poor quality and 10 being excellent, and there were separate datasets for red and white wine.
The dataset has already been simplified in scope of this analysis, merging together the red and white wine datasets and dividing the wines in only two categories; they are either assigned to category 0 (low quality, original score <= 5), or category 1 (high quality, original score => 7). Wines with grade 6 have been removed, to simplify the already fairly complex classification task.

The properties taken into account are the following, and will be referred to with their order number:

0 - Fixed acidity
1 - Volatile acidity
2 - Citric acid
3 - Residual sugar
4 - Chlorides
5 - Fee sulfur dioxide
6 - Total sulfur dioxide
7 - Density
8 - pH
9 - Sulphates
10 - Alcohol

We'll start the analysis with some exploratory data analysis, to verify the distribution of values in the features and to discern which features may be the most suited for our classification task.

Univariate analysis

We are now going to analyze each feature by its properties and plotting histograms and boxplots using the *matplotlib* library. The following metrics have been utilized:

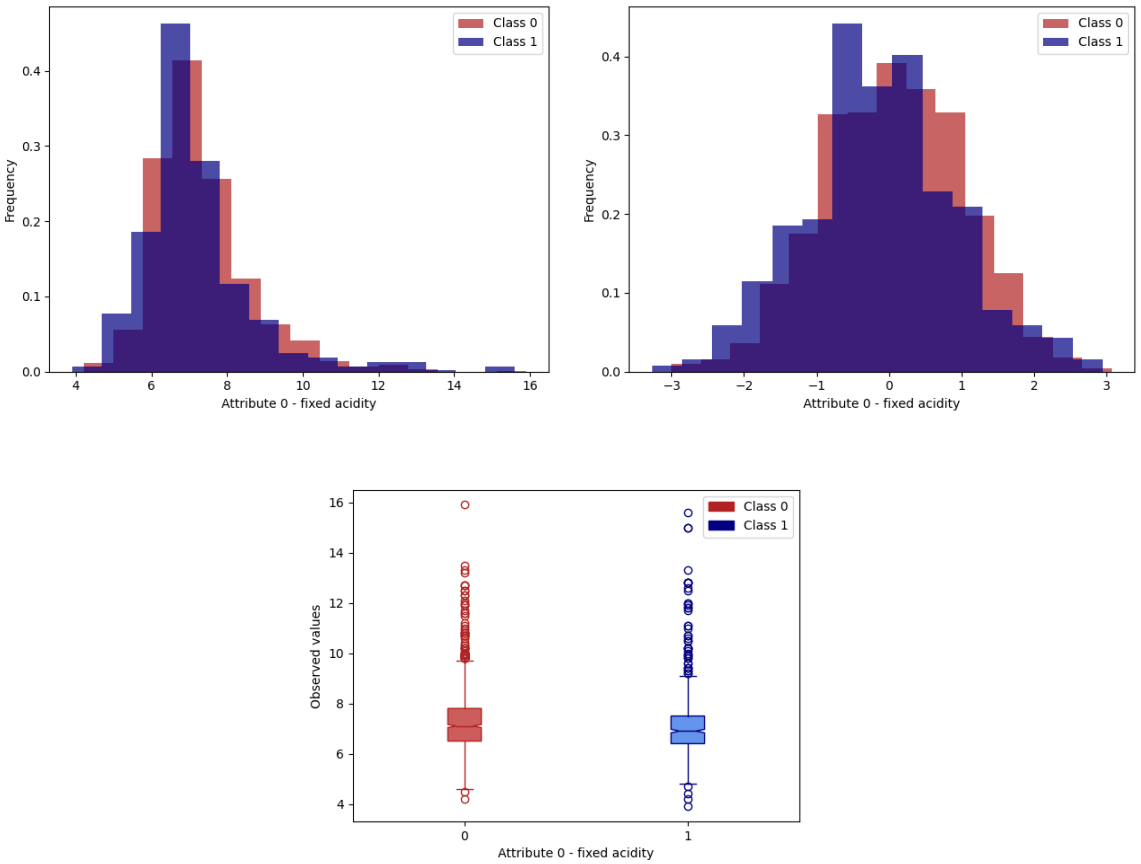
- mean
- class conditional mean
- variance
- skewness (Fisher - Pearson coefficient of skewness, found in *scipy.stats* module)

Two different kinds of preprocessing have been applied on the data. *Gaussianization* has been performed on each feature to help with Gaussian classifiers and to reduce the

impact of the outliers on the dataset. The operation has been performed by assigning a rank to each sample for each feature, dividing the rank by the number of features and then numerically computing a percent point function (inverse of the probability density function) to the output. The function has been computed with the *ppf* function from the *scipy.stats.norm* module.
After the univariate analysis we will consider Principal Component Analysis to verify the effect of dimensionality reduction on the classifier results.

Feature 0 - Fixed acidity

Acidity is a fundamental property of wine, imparting sourness and resistance to microbial infection.
The minimum value is **3.9**, the maximum value is **15.9**.
The mean of all samples is **7.258809**.
Class conditional mean for class 0 is **7.310033**, and computed for class 1, is **7.156362**.
The variance is **1.833653**, whereas skewness is **2.556713**.



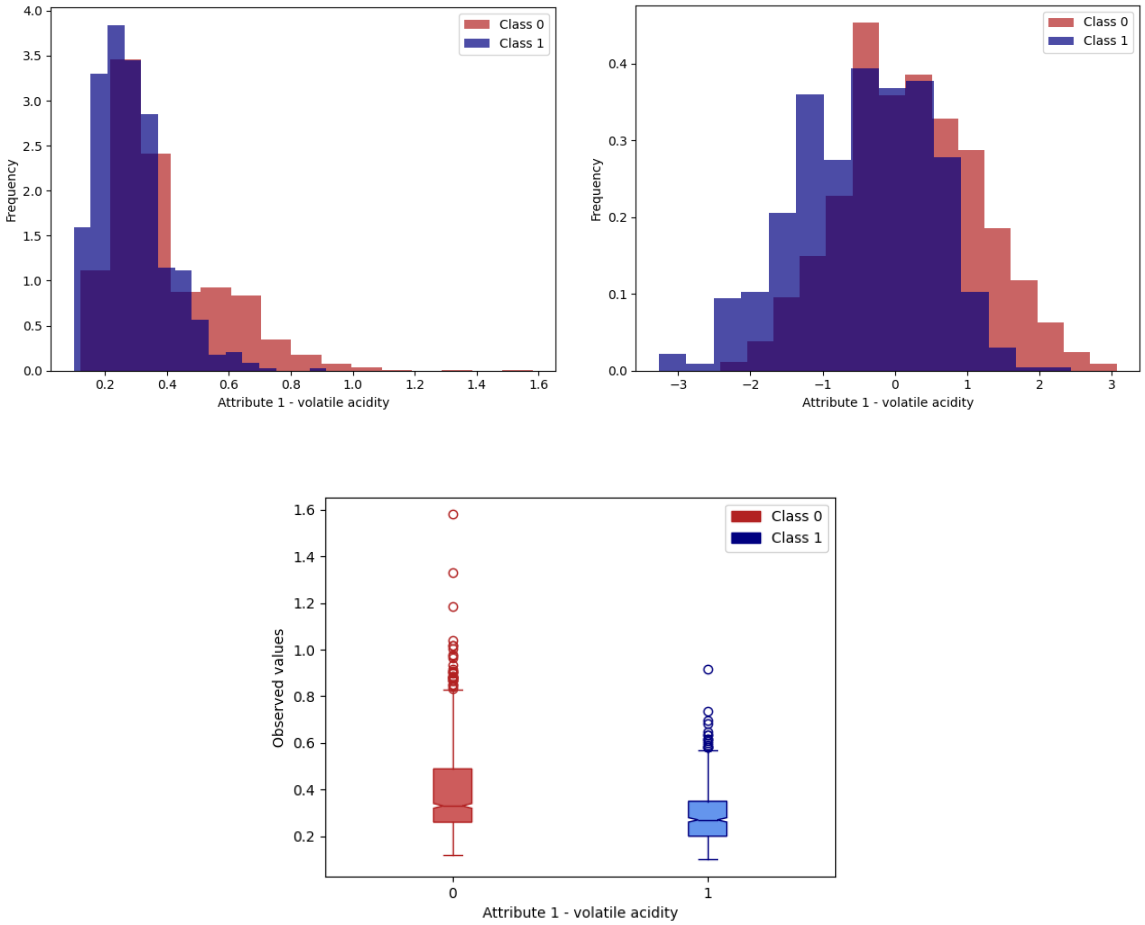
Normalized histogram of the feature on the left, histogram of the gaussianized feature on the right.
Boxplot central line shows where the mean is.

Feature 0 shows a fairly regular normal distribution for both classes, with a large presence of outliers in the right part of the histogram, as also shown in the boxplots.
Class conditional means are fairly similar, with class 1 having a lower mean and a higher "peak" at the mean.

This feature is probably not going to be determinant in the classification problem, due to the high level of overlapping of the two classes.

Feature 1 - Volatile acidity

Volatile acidity is a measure of the low molecular weight (or steam distillable) fatty acids in wine and is generally perceived as the odour of vinegar.
The minimum value is **0.1**, the maximum value is **1.58**.
The mean of all samples is **0.354905**.
Class conditional mean for class 0 is **0.388985**, and computed for class 1, is **0.286746**.
The variance is **0.028979**, whereas skewness is **2.569204**.
.

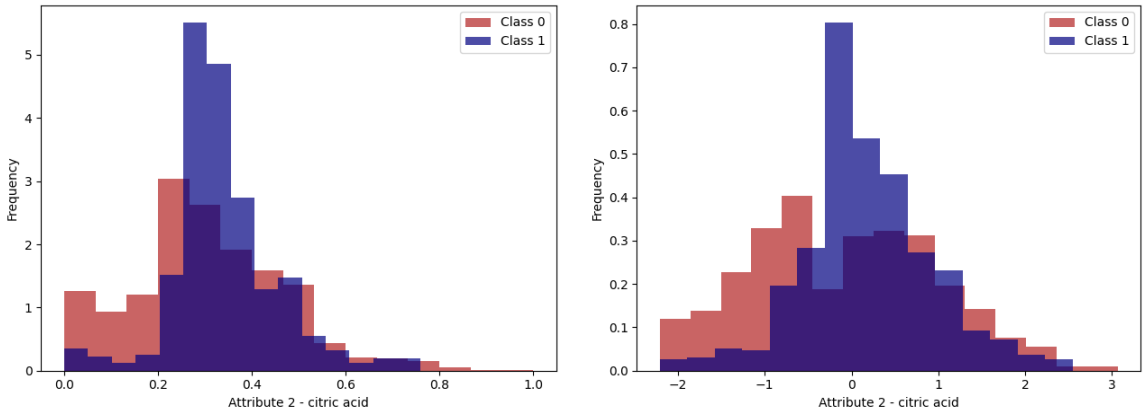


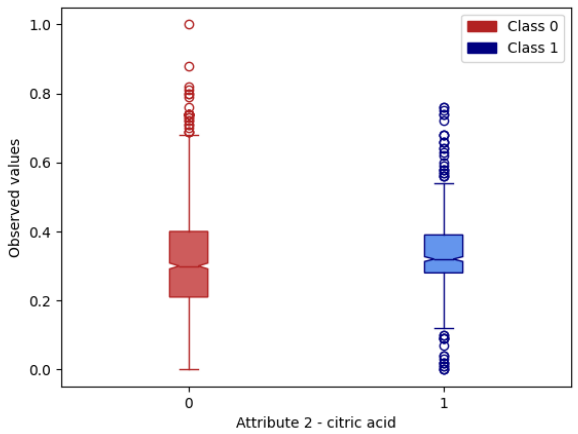
Normalized histogram of the feature on the left, histogram of the gaussianized feature on the right.
Boxplot central line shows where the mean is.

Volatile acidity seems to be much more helpful in the classification task: class 0 seems to have a much higher variance and skewness to the right, with a handful of outliers showing higher values than the average. Class conditional means are fairly different, as visible in the boxplots, and the higher values of class 0 is relevant in our quest.
It's already worth noting that even though they are both acidity measures, fixed and volatile acidity seem to have very little to none correlation.

Feature 2 - Citric Acid

The citric acid most commonly found in wine is commercially produced acid supplements derived from fermenting sucrose solutions. These inexpensive supplements can be used by winemakers in acidification to boost the wine's total acidity.
The minimum value is **0**, the maximum value is **1**.
The mean of all samples is **0.316259**.
Class conditional mean for class 0 is **0.303891**, and computed for class 1, is **0.340995**.
The variance is **0.021977**, whereas skewness is **2.398319**.



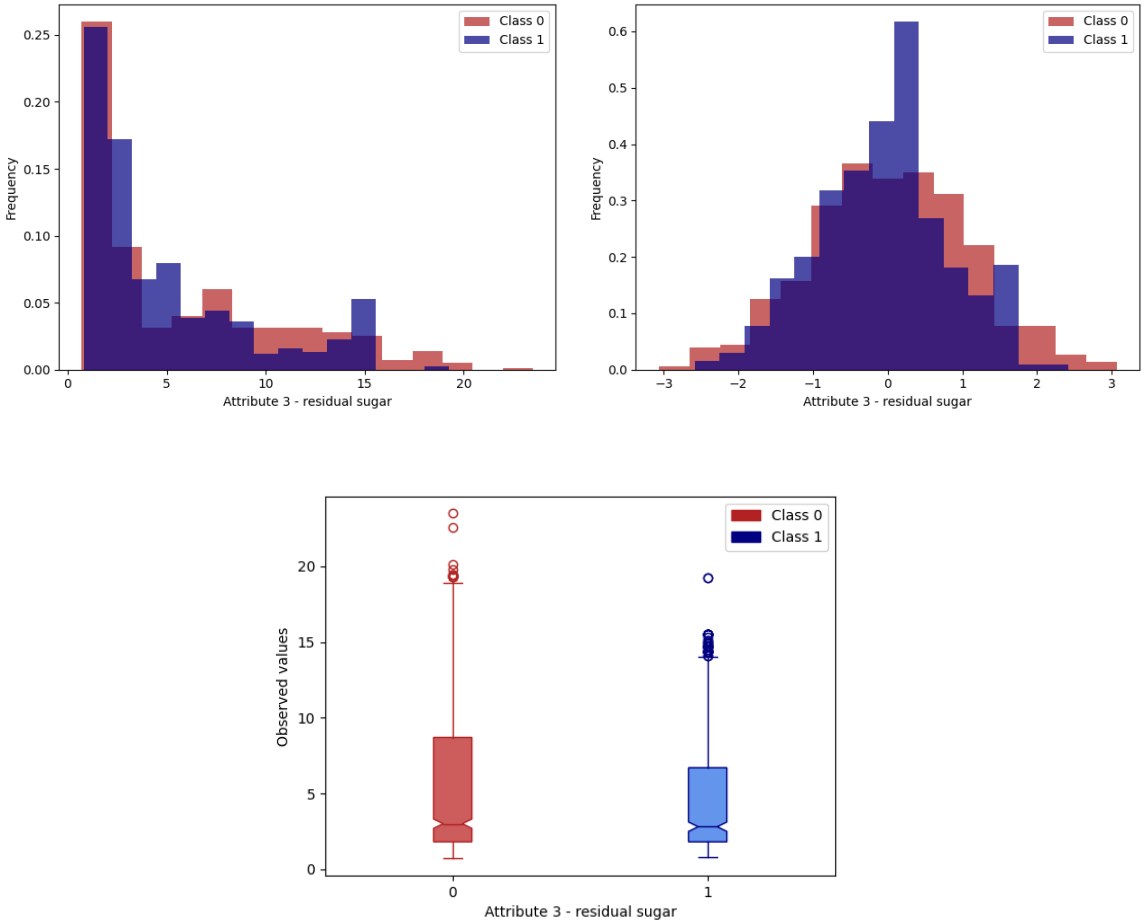


Normalized histogram of the feature on the left, histogram of the gaussianized feature on the right.
Boxplot central line shows where the mean is.

Citric acid optimal levels for higher quality wine seem to be around the dataset mean of 0.316259. The lower spectrum of the distribution seems to be filled with lower quality wines. Also higher values of the spectrum relate more to class 0 than to class 1, making this feature interesting for classification purposes.

Feature 3 - Residual Sugar

Residual Sugar is from natural grape sugars leftover in a wine after the alcoholic fermentation finishes.
The minimum value is **0.7**, the maximum value is **23.5**.
The mean of all samples is **5.442197**.
Class conditional mean for class 0 is **5.712928**, and computed for class 1, is **4.900734**.
The variance is **22.511664**, whereas skewness is **2.557445**.

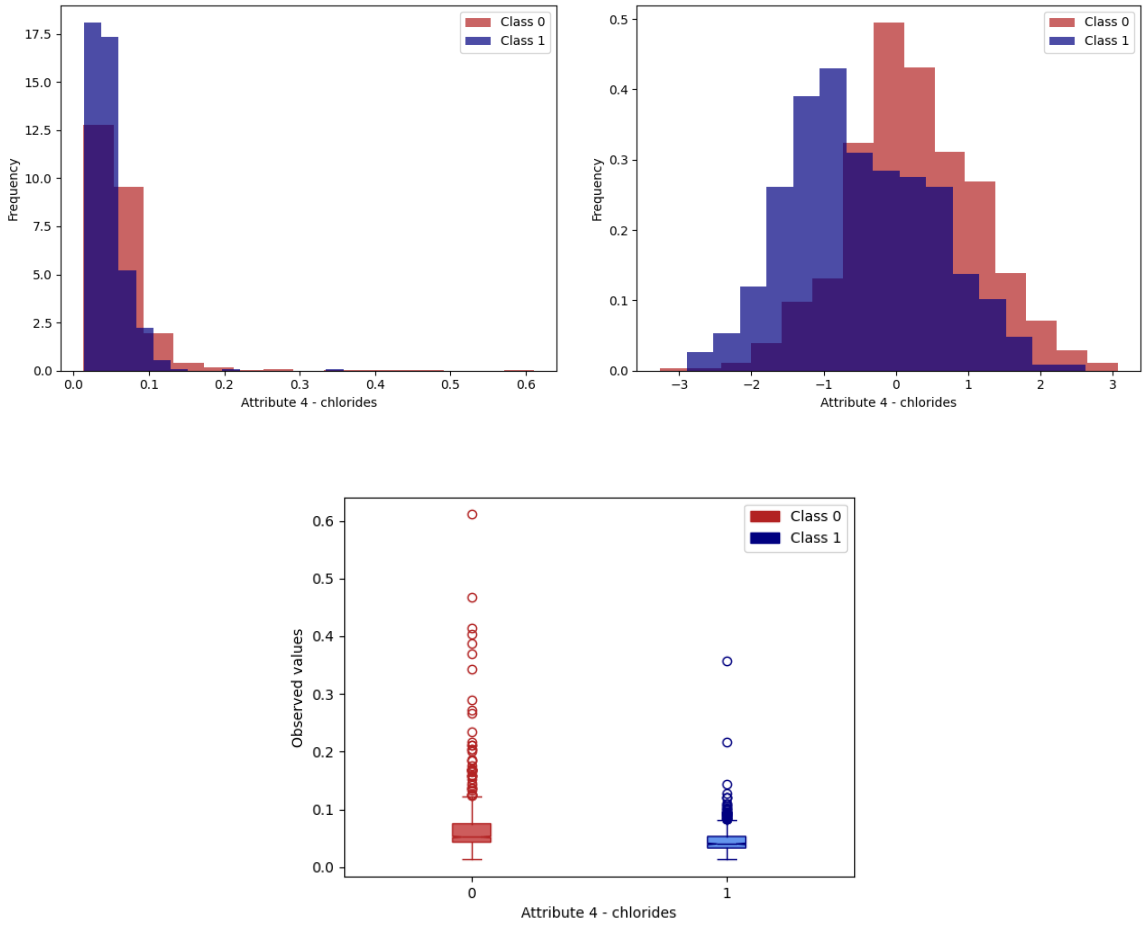


Normalized histogram of the feature on the left, histogram of the gaussianized feature on the right.
Boxplot central line shows where the mean is.

The residual sugar feature shows a very different behavior from an ideal normal distribution: the mean is very low and we have values very far from that value. Gaussianization helps a lot with the visualization of the tendency that we have already seen in other features: class 1 has a very clear “ideal value”, with a very defined mean, whereas class 0 has a more spread-out distribution, in particular towards the higher end of the spectrum.

Feature 4 - Chlorides

The amount of chloride in wine is influenced by both the terroir and type of grape, and the wine flavor is strongly impacted by this particular ion, which gives the wine an undesirable salty taste.
The minimum value is **0.013**, the maximum value is **0.611**.
The mean of all samples is **0.057574**.
Class conditional mean for class 0 is **0.063165**, and computed for class 1, is **0.046393**.
The variance is **0.001376**, whereas skewness is **2.660304**.

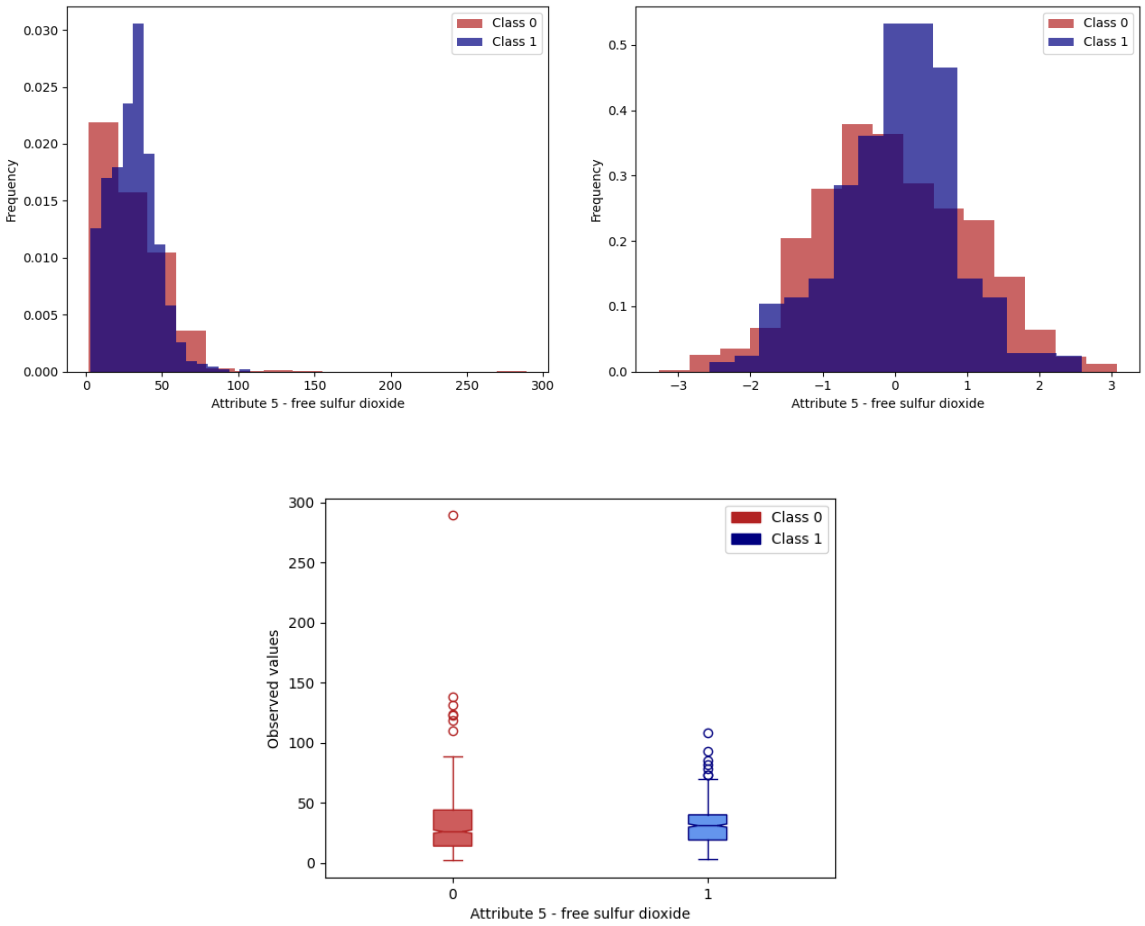


Normalized histogram of the feature on the left, histogram of the gaussianized feature on the right.
Boxplot central line shows where the mean is.

The amount of Chlorides in the analysis seems in fact to be inversely related to the perceived wine quality; class 0 has higher chlorides values, with both mean and outliers more spread out on the right part of the histograms.
It will probably be a great feature for classification purposes.

Feature 5 - Free Sulfur Dioxide

Sulfur dioxide is important in the winemaking process as it aids in preventing microbial growth and the oxidation of wine. The free sulfites are those available to react and thus exhibit both germicidal and antioxidant properties.
The minimum value is **2**, the maximum value is **289**.
The mean of all samples is **30.146003**.
Class conditional mean for class 0 is **30.094209**, and computed for class 1, is **30.249592**.
The variance is **369.534556**, whereas skewness is **2.738651**.

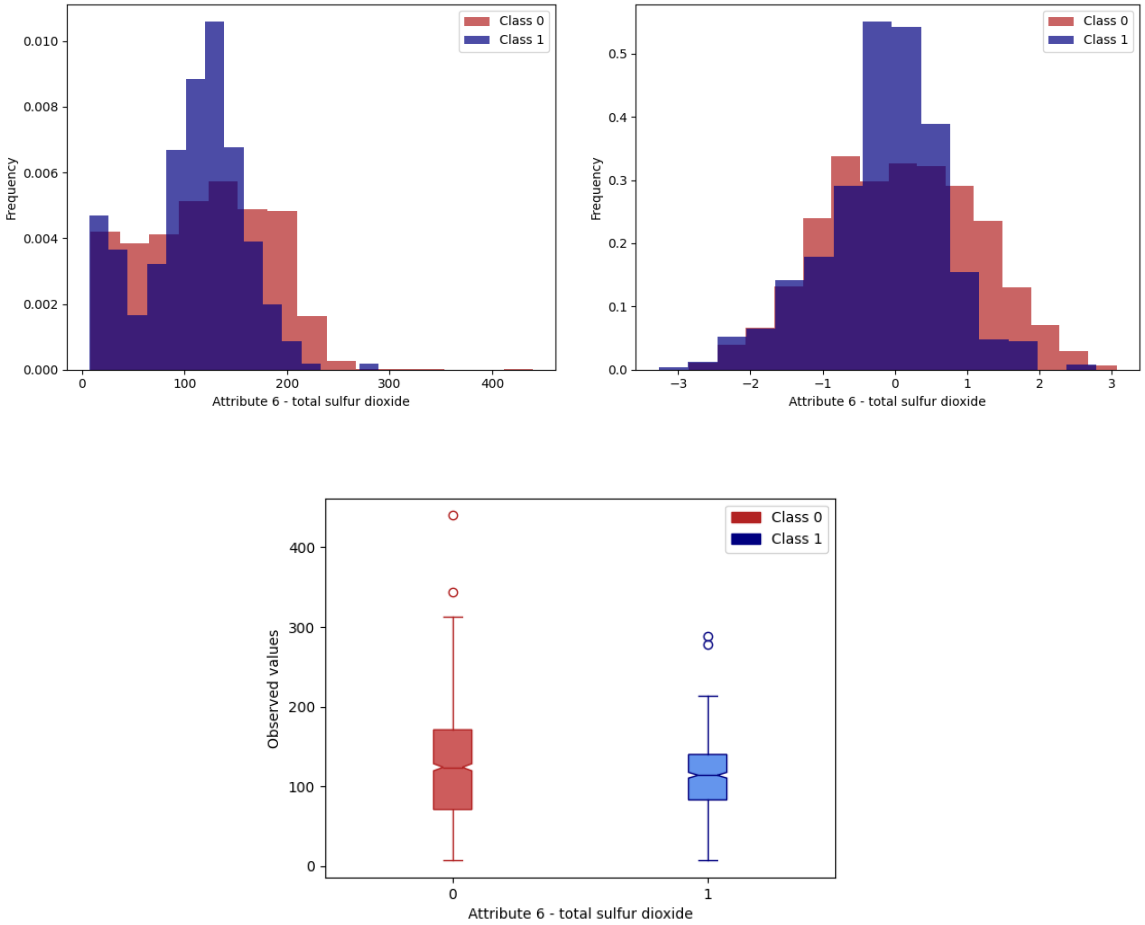


Normalized histogram of the feature on the left, histogram of the gaussianized feature on the right.
Boxplot central line shows where the mean is.

Free sulfur dioxide measurements prove to be interesting especially for class 1, with a very high frequency around the mean value, even though the class conditional means are fairly similar.
Class 0 shows a higher number of outliers, and is more skewed to the right.

Feature 6 - Total Sulfur Dioxide

Total sulfites include free and bound Sulfur Dioxide. The bound sulfites are those that have reacted with other molecules within the wine medium.
The minimum value is **7**, the maximum value is **440**.
The mean of all samples is **116.349918**.
Class conditional mean for class 0 is **120.232463**, and computed for class 1, is **108.584829**.
The variance is **3374.391287**, whereas skewness is **2.409429**.

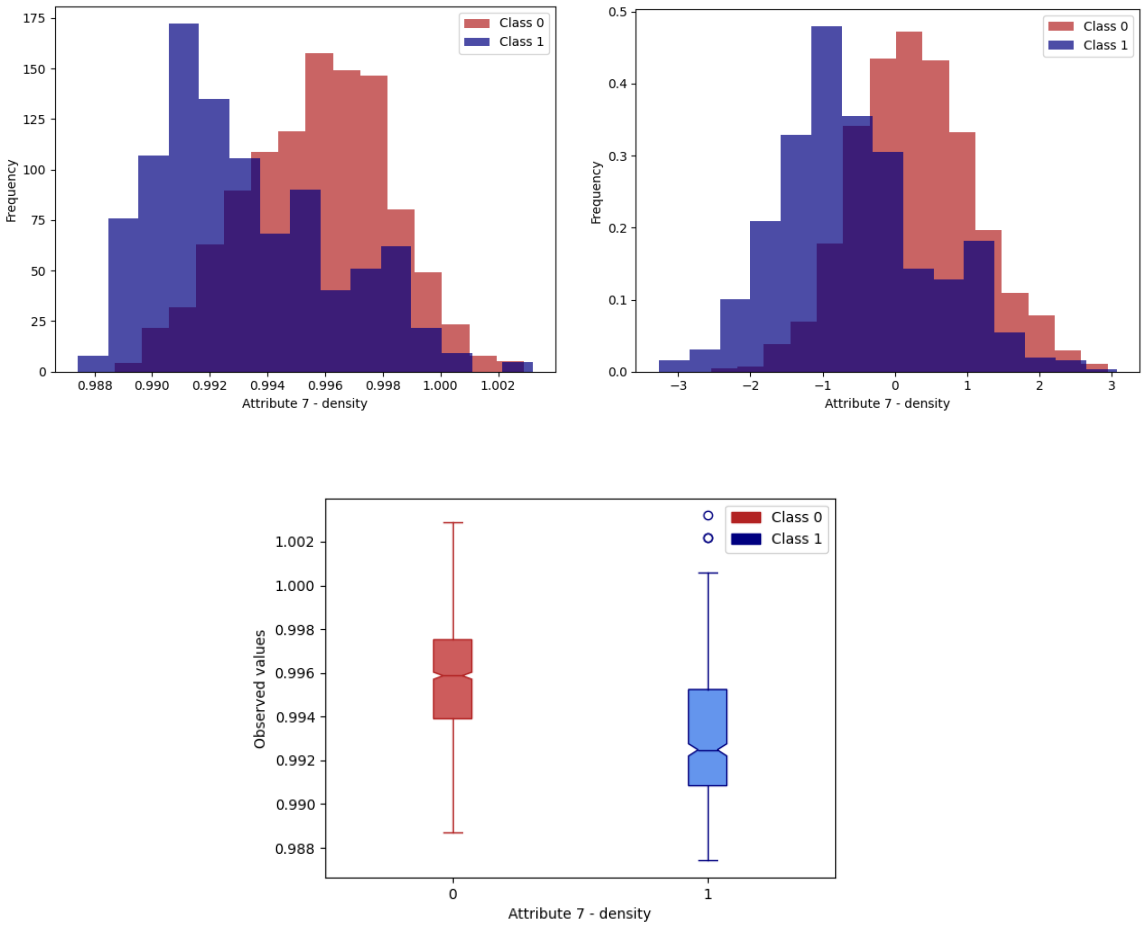


Normalized histogram of the feature on the left, histogram of the gaussianized feature on the right.
Boxplot central line shows where the mean is.

Total sulfur dioxide values show a high level of correlation with free sulfur dioxide; the same characteristics of class 1 and class 0 are present, with a high peak around the mean for the former and a skew to the right for the latter.

Feature 7 - Density

The minimum value is **0.987420**, the maximum value is **1.003200**.
The mean of all samples is **.994863**.
Class conditional mean for class 0 is **0.995707**, and computed for class 1, is **0.993175**.
The variance is **0.000009**, whereas skewness is **2.752156**.

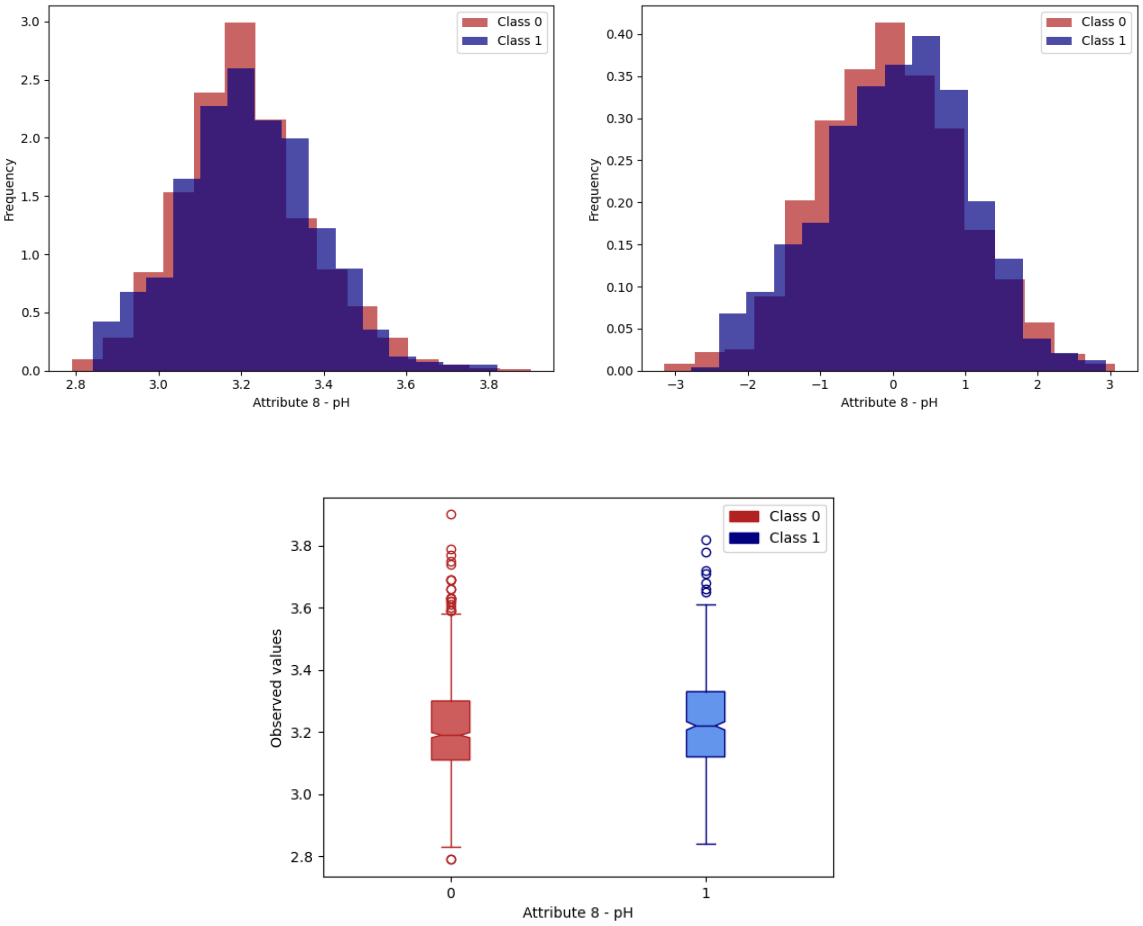


Normalized histogram of the feature on the left, histogram of the gaussianized feature on the right.
Boxplot central line shows where the mean is.

Despite the extremely low variance and the reduced range of values, density proves to be a very interesting feature for classification purposes, with a clear division of the classes distributions. Even before gaussianization the feature provides a strong gaussian behavior, and it's clearly going to be a decisive feature for classification.

Feature 8 - pH

The minimum value is **2.79**, the maximum value is **3.9**.
The mean of all samples is **3.213953**.
Class conditional mean for class 0 is **3.211085**, and computed for class 1, is **3.219690**.
The variance is **0.025403**, whereas skewness is **2.682355**.

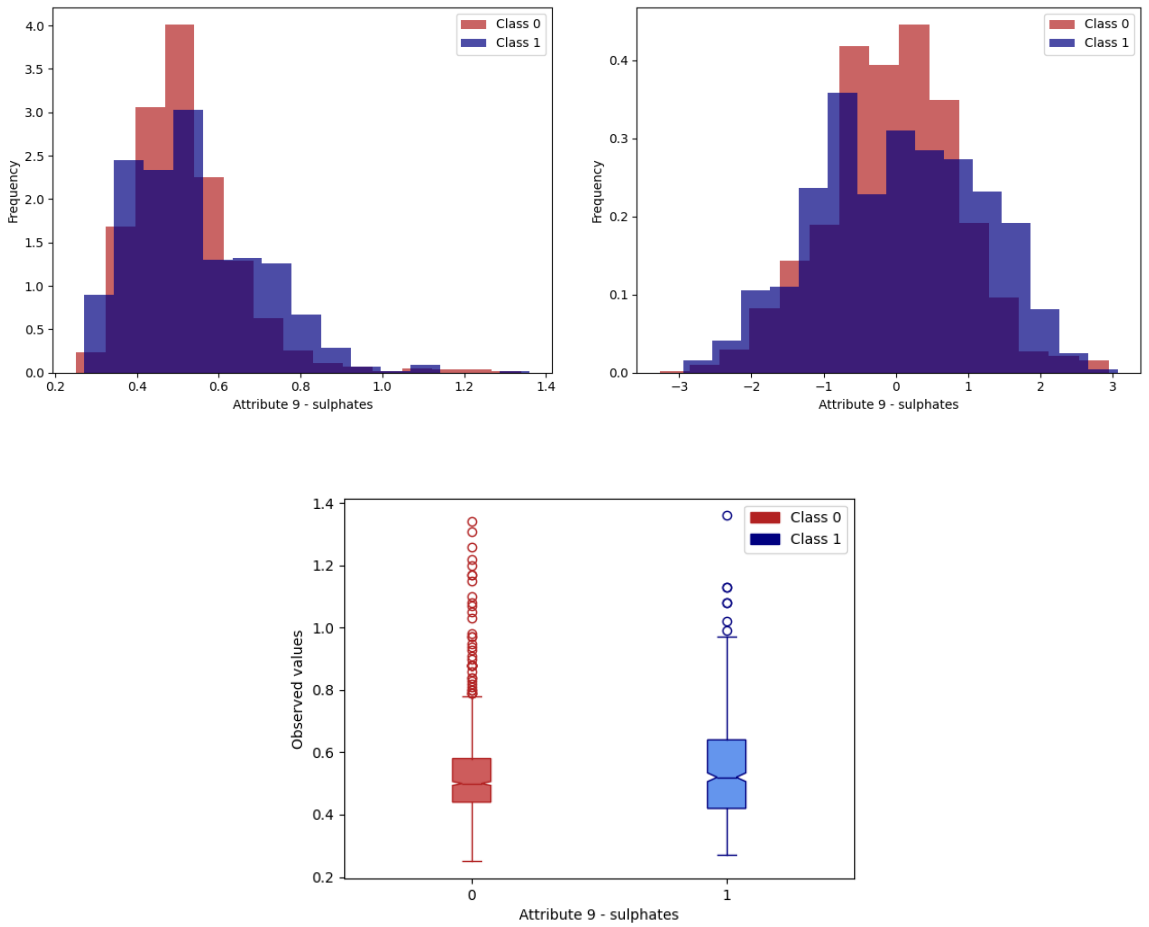


Normalized histogram of the feature on the left, histogram of the gaussianized feature on the right.
Boxplot central line shows where the mean is.

Feature 8, the wine pH, shows a fairly regular normal distribution, without significative differences between classes. It doesn't seem it will prove particularly useful for classification purposes, due to very poor class division.

Feature 9 - Sulphates

The minimum value is **0.25**, the maximum value is **1.36**.
The mean of all samples is **0.528434**.
Class conditional mean for class 0 is **0.521158**, and computed for class 1, is **0.542985**.
The variance is **0.020420**, whereas skewness is **2.711495**.

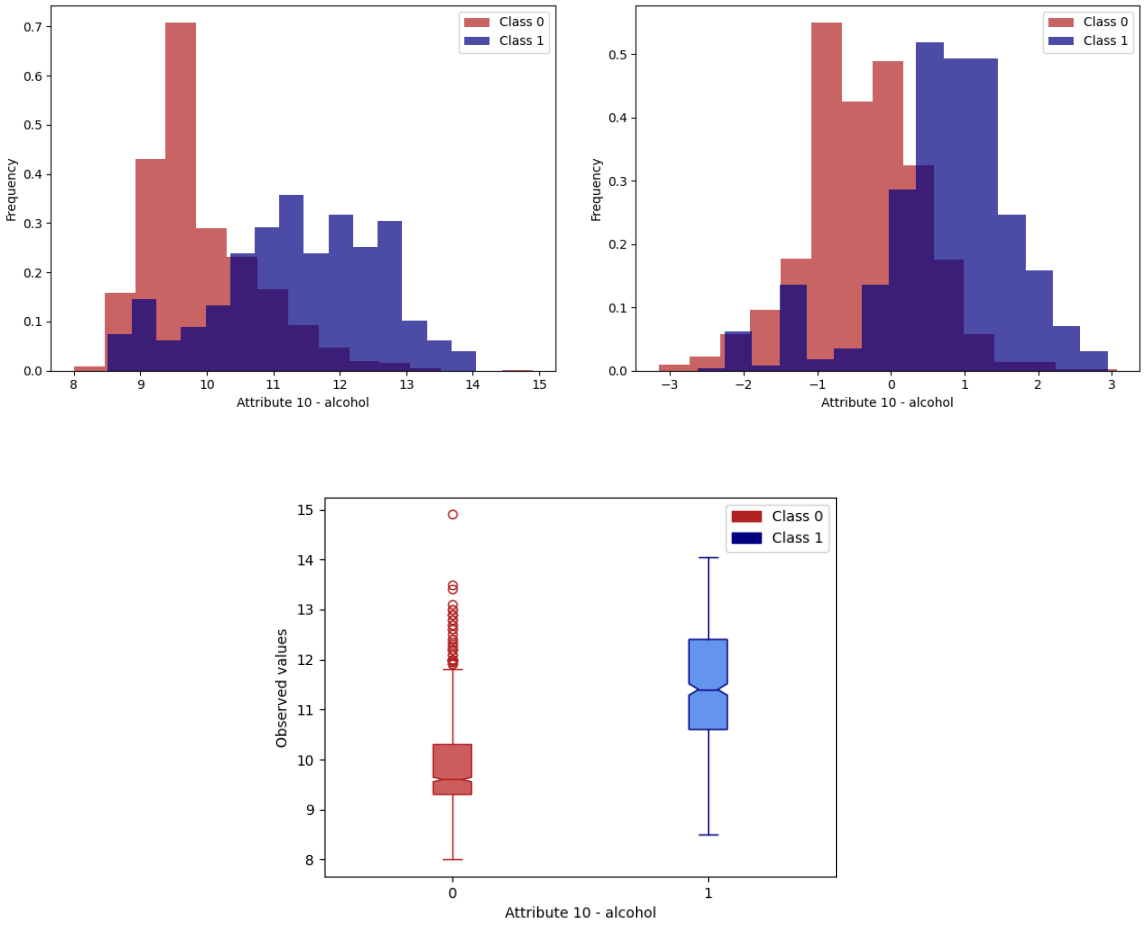


Normalized histogram of the feature on the left, histogram of the gaussianized feature on the right.
Boxplot central line shows where the mean is.

Feature 9, amount of sulphates, proves to be fairly interesting to the more right skewed distribution of class 1, and to the higher central tendency around the mean of class 0. It could have good performance with gaussian classifiers, making it an interesting feature for the task.

Feature 10 - Alcohol

The minimum value is **8**, the maximum value is **14.9**.
The mean of all samples is **10.379824**.
Class conditional mean for class 0 is **9.878015**, and computed for class 1, is **11.383442**.
The variance is **1.500919**, whereas skewness is **2.642996**.

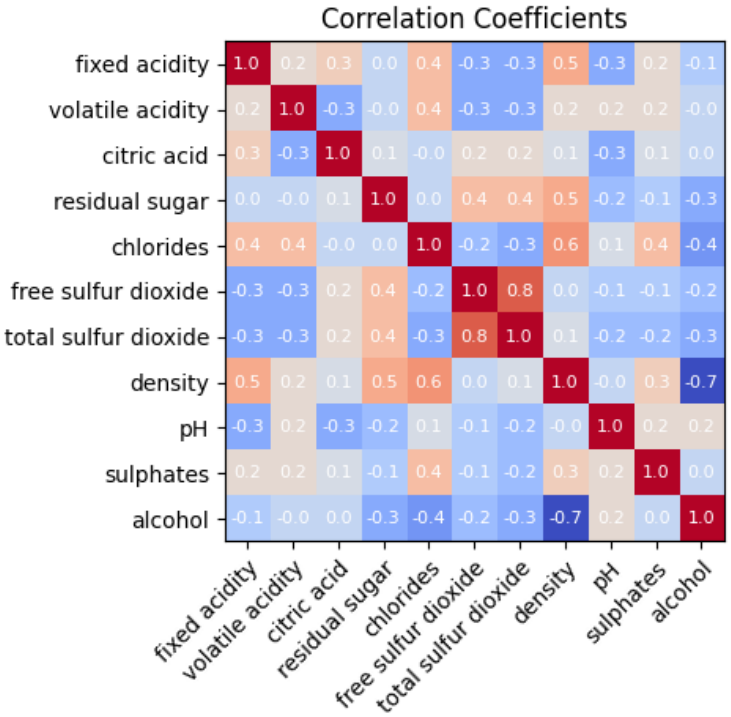


Normalized histogram of the feature on the left, histogram of the gaussianized feature on the right.
Boxplot central line shows where the mean is.

Feature 10, alcohol level, is probably the most interesting feature; there is clear evidence that higher alcohol levels tend to yield higher perceived wine quality. With a difference in class conditional mean as significant as feature overall variance this feature shows very interesting behavior.
It could be determinant in discerning classes in our models.

Correlation analysis

The computation of the correlation matrix was helpful to better grasp relationships between the features, and to evaluate possible high-correlation couples. The matrix was computed through the *numpy.corrcoef* method, exploiting the Pearson product moment correlation coefficients.



The graph shows correlation and anti-correlation between features. The highest correlation is between free and total sulfur dioxide features; this behavior makes sense in the domain, and it had already shown in the histograms, where the features had very similar distributions.
Another interesting couple is alcohol and density; the anti-correlation between those two features makes sense from a chemical point of view, and from the histograms we can see the opposite behavior of class 0 and class 1 in the analysis of those features.
Density is also fairly strongly correlated with chloride presence. Also this phenomenon makes sense in the domain context, with chlorides having a higher density than water, and could prove interesting.

Dimensionality reduction

After the analysis of correlation and anti-correlation between features it is reasonable to analyze if any kind of dimensionality reduction may result useful for our task.
In this case we'll perform Principal Component Analysis on the dataset, with two different numbers of components extracted.
With 11 starting features, we'll test a m=10 PCA and a m=5 PCA. With the m=5 being quite

extreme we are verifying the result of heavy reduction on the accuracy of our classification models, whereas we expect interesting results from our m=10 reduction, also in light of the correlation between some of the features.

Multivariate Gaussian Classifier

The first classifier of our analysis will be a generative gaussian classifier. Through the computation of class conditional means and covariance matrices related to our training dataset we'll produce scores in terms of class posteriors with our test dataset.

All the preliminary analysis on the classifiers will be performed using k-fold cross validation with k=5, thanks to the *scikit-learn KFold* method, which provides the indices representing the folds to use as train and test. We will use our test dataset as the final evaluation dataset.

We will test base MVG, with computation of mean and complete covariance matrix for each class, as well as Naive Bayes MVG, Tied Covariance, and a combination of both.
With Naive Bayes we are treating the dataset as all linearly independent features, with a diagonal covariance Matrix, and with the Tied Covariance hypotheses we are computing a single covariance matrix with both classes sharing it.

Due to our classes often natively showing a fairly regular normal distribution we expect the classifier to have a good performance; our metric for it will be the minimum Detection Cost Function value, so we will perform a scores calibration in order to find the best possible threshold for our application.
We will also test some various possible applications with varying priors and False Negative/False positive costs, while considering a balanced application as our main one.

Multivariate Gaussian Classifier					
	$\pi(0.5), cfn(1), cfp(1)$	$\pi(0.8), cfn(1), cfp(1)$	$\pi(0.2), cfn(1), cfp(1)$	$\pi(0.5), cfn(10), cfp(1)$	$\pi(0.5), cfn(1), cfp(10)$
Raw	**0.358**	0.608	0.648	0.751	0.788
Gauss	0.370	0.574	0.668	0.690	0.805
Pca10	0.362	0.616	0.654	0.761	0.789
Pca5	0.408	0.757	0.691	0.921	0.833
Pca10 - Gauss	0.360	0.568	0.646	0.670	0.782

The classifier has the best performance over our main application, with balanced priors. We can see that we obtain the best performance over raw features, but PCA with m=10 seems to be very effective, with extremely close results. Also PCA scores better than the raw features over all other applications, and PCA over gaussianized features achieves even lower costs.

The effect of PCA doesn't surprise us due to the correlation between features analyzed before, so these results are coherent with what we expected.

Naive Bayes Multivariate Gaussian Classifier					
	π(0.5), cfn(1), cfp(1)	π(0.8), cfn(1), cfp(1)	π(0.2), cfn(1), cfp(1)	π(0.5), cfn(10), cfp(1)	π(0.5), cfn(1), cfp(10)
Raw	0.414	0.778	0.655	0.839	0.794
Gauss	0.456	0.824	0.692	0.858	0.843
Pca10	**0.405**	0.725	0.747	0.881	0.858
Pca5	0.416	0.854	0.736	0.959	0.853
Pca10 - Gauss	**0.405**	0.716	0.693	0.784	0.850

Naive Bayes MVG scores generally lower than normal MVG, but confirms the efficacy of PCA with m=10, which scores better than raw and gaussianized features. It is also worth noting that gaussianized and dimensionality reduced features score better than the rest in most applications.

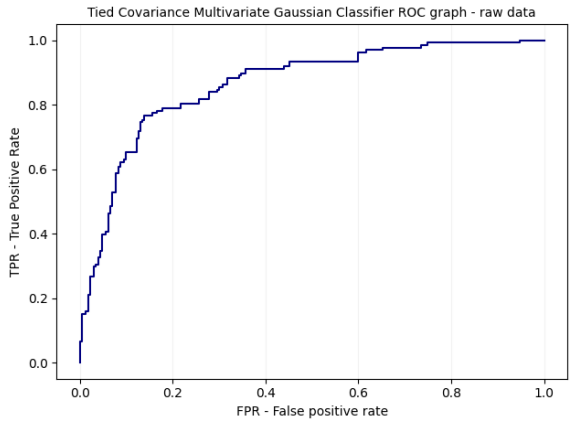
Tied Covariance Multivariate Gaussian Classifier					
	π(0.5), cfn(1), cfp(1)	π(0.8), cfn(1), cfp(1)	π(0.2), cfn(1), cfp(1)	π(0.5), cfn(10), cfp(1)	π(0.5), cfn(1), cfp(10)
Raw	**0.335**	0.659	0.619	0.708	0.787
Gauss	0.365	0.641	0.624	0.788	0.809
Pca10	0.339	0.620	0.633	0.691	0.804
Pca5	0.413	0.752	0.679	0.842	0.840
Pca10 - Gauss	0.342	0.604	00.637	0.688	0.802

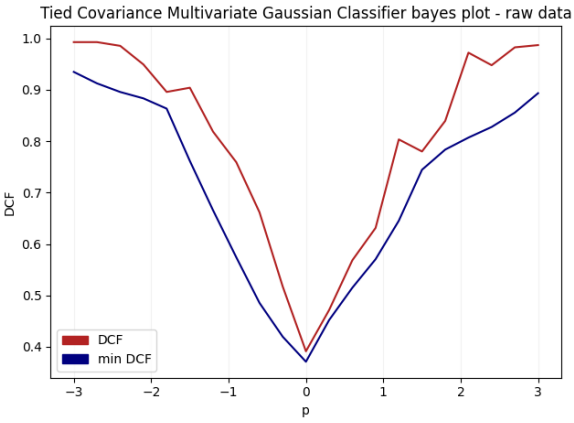
Tied covariance MVG performs extremely good, in particular on our main application. It achieves the lowest up to now scores of all the classifiers previously tried. It's possible that the higher amount of data over which the shared covariance matrix is computed helps us have higher accuracy in our prediction.
It's also worth noting that the raw features scored best, even though the PCA reduced datasets are not far from that result.

Naive Bayes Tied Covariance Multivariate Gaussian Classifier					
	π(0.5), cfn(1), cfp(1)	π(0.8), cfn(1), cfp(1)	π(0.2), cfn(1), cfp(1)	π(0.5), cfn(10), cfp(1)	π(0.5), cfn(1), cfp(10)
Raw	0.406	0.802	0.660	0.888	0.864
Gauss	0.415	0.810	0.718	0.910	0.850
Pca10	**0.343**	0.627	0.662	0.717	0.824
Pca5	0.423	0.758	0.692	0.867	0.848
Pca10 - Gauss	0.351	0.626	0.654	0.704	0.842

The combined Naive Bayes and Tied Covariance hypothesis yield a model where PCA shows the best results. Both the gaussianized and non gaussianized version show better scores over all the considered applications.

Summing up, our best performing model is the Tied Covariance MVG over raw features, closely followed by MVG with PCA. In general PCA seems to have a positive effect on the accuracy of the classification, and the classifier generally seems to be fairly useful even in unbalanced applications.





We can see the efficacy of the model on the varying thresholds on the ROC graph on the left; the Bayes Plot on the right shows us that the scores are fairly uncalibrated on the extremes of our possible applications, but pretty calibrated around the balanced one.

Logistic Regression Classifier

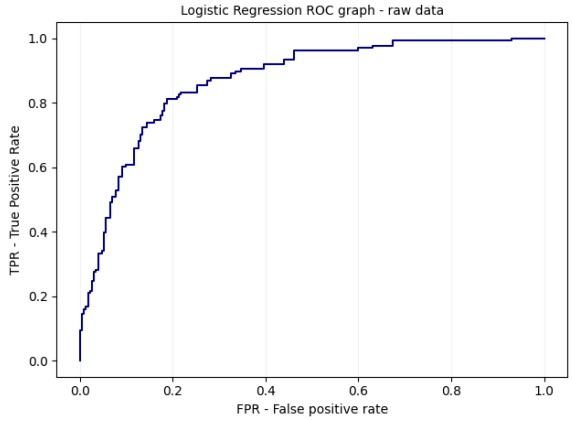
The next classifier we'll test is the Logistic Regression classifier. The results are not log-likelihoods but directly class posteriors, or better scores that discriminate between class 0 and class 1.
It is numerically approximated through the minimization of the Logistic Loss function.

We'll also perform *5-fold-cross-validation* to find a suitable value for the regularization term *lambda*. 17 values between *1e-8* and 1 have been tested. We considered the performance on the main application using raw data as our evaluation metric. Between the tested value the best performing lambda is *1e-4*, so that will be our value of choice.

Logistic Regression					
	$\pi(0.5), C_{fn}(1), C_{fp}(1)$	$\pi(0.8), C_{fn}(1), C_{fp}(1)$	$\pi(0.2), C_{fn}(1), C_{fp}(1)$	$\pi(0.5), C_{fn}(10), C_{fp}(1)$	$\pi(0.5), C_{fn}(1), C_{fp}(10)$
Raw	**0.336**	0.554	0.651	0.633	0.815
Gauss	0.359	0.610	0.654	0.727	0.813
Pca10	0.338	0.549	0.646	0.625	0.808
Pca5	0.416	0.747	0.686	0.838	0.842
Pca10 - Gauss	0.343	0.538	0.642	0.617	0.804

The model performs almost as good as the best MVG classifiers on our main application. It even achieves better scores on some of our other applications, possibly making it a more flexible candidate if we are not sure about the kind of use our model will have.

It's also worth noting that the performance on PCA and gaussianized data is the best on alternative applications, even though it loses some points on the balanced one. This doesn't surprise us as the whitening and normalization of the data normally positively affects the results of the classifier.



From the Bayes plot with data of one of the folds we can see that the scores are slightly uncalibrated but consistently along the application range. It's possible that a minor correction of the threshold helps us with this.

Support Vector Machine Classifier

The Support Vector Machine classifier is also cast as a minimization problem, looking for the separation hyperplane between the classes. This classifier does not provide scores with probabilistic interpretation, but a positive score if the sample is predicted as class 1 and a negative one in case of class 0.

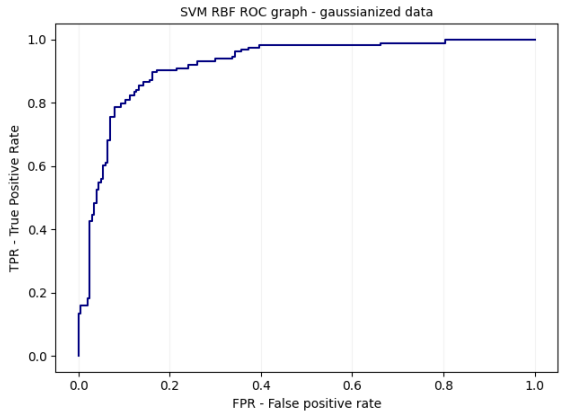
We will test standard SVM, and then we will use various kernel functions to expand the features to a higher-dimensional space, and see if they benefit from a non-linear separation plane. We will test only our main application, since it is harder to incorporate different priors in the SVM scores.

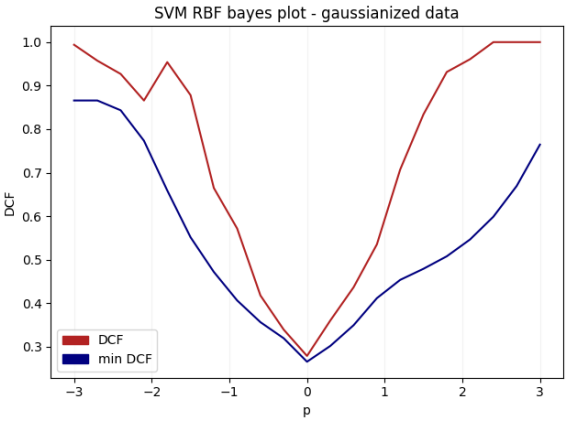
Also in this case we will use *5-fold-cross-validation* and all the model's hyperparameters will be tuned this way. There are quite a few, and we'll report the best performing values here:

C (bounds term for the minimization function) = 1
K (regularization term for the extended feature matrix) = 1
Gamma (*RBF Kernel* exponential term) = 0.1
d (*Polynomial Kernel* degree) = 3
c (constant in the Polynomial Kernel) = 2

Support Vector Machine Classifier			
	Base	Kernel = RBF	Kernel = Poly
Raw	0.350	0.652	0.673
Gauss	0.357	0.294	0.507
Pca10	0.386	0.653	0.707
Pca5	0.515	0.657	0.668
Pca10 - Gauss	0.339	0.294	0.477

As we can see the classifier shows excellent results, in particular in non-linear feature spaces and with regularization of the features through Gaussianization.
The Radial Basis Function Kernel is by far the best performing, and the performance difference between raw and regularized features is particularly evident. In general the classifier seems not to particularly benefit from dimensionality reduction through PCA.





As we can see from the Bayes plot from one of the folds of cross validation, the scores are well calibrated around our main application point, even though they are pretty far at the extremes of the priors.
We can consider this model the best performing up to now, and also consider our threshold already pretty accurate.

Gaussian Mixture Model Classifier

The last classifier we'll consider is a GMM classifier, where different gaussian models are combined as components, with every one of them having a weight.
The single MVG classifier had a good performance, so we expect this model to perform similarly.

We will test three versions of the model, as per the MVG; the base one, the Naive Bayes one with diagonal covariance matrices and the Tied Covariance matrix version of the model.

To avoid degenerate solutions we have constrained the covariance matrix eigenvalues to be higher or at most equal than a parameter *psi,* which has been set to 0.01.
The number of components has been set to 64, iterating the *LBG* step of the GMM components estimation 6 times.
This constraint is due to the fairly high estimation time needed, which is multiplied by the 5-fold-cross-validation approach which has been kept to remain consistent with other model's analysis.

Gaussian Mixture Model Classifier					
	π(0.5), cfn(1), cfp(1)	π(0.8), cfn(1), cfp(1)	π(0.2), cfn(1), cfp(1)	π(0.5), cfn(10), cfp(1)	π(0.5), cfn(1), cfp(10)
Raw	**0.364**	0.632	0.653	0.747	0.767
Gauss	0.534	0.816	0.821	0.899	0.877
Pca10	0.368	0.640	0.658	0.762	0.770
Pca5	0.527	0.859	0.806	0.968	0.879
Pca10 - Gauss	0.497	0.708	0.771	0.771	0.870

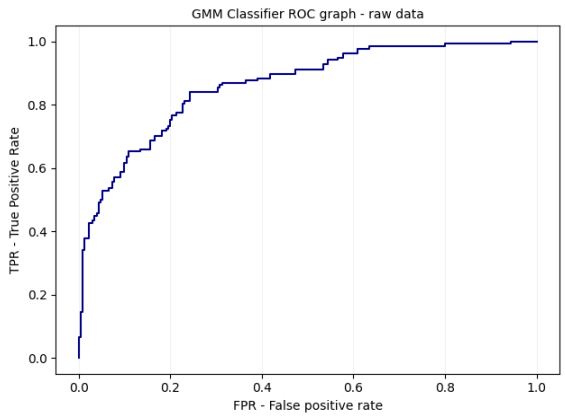
The base GMM classifier performs similarly to the MVG classifier, even though it has generally lower scores on our main application.
It is possible that adding components to the model helps it achieve lower detection costs, but in previous performed analysis of this model the increase of components didn't particularly boost the model performance, if not for some small percentages.

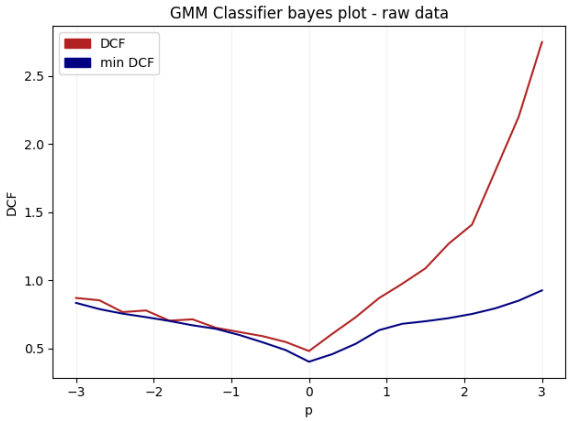
Gaussian Mixture Model Naive Bayes Classifier					
	π(0.5), cfn(1), cfp(1)	π(0.8), cfn(1), cfp(1)	π(0.2), cfn(1), cfp(1)	π(0.5), cfn(10), cfp(1)	π(0.5), cfn(1), cfp(10)
Raw	0.589	0.859	0.871	0.911	0.954
Gauss	**0.504**	0.782	0.738	0.866	0.877
Pca10	0.604	0.936	0.857	0.990	0.933
Pca5	0.591	0.963	0.831	0.998	0.904
Pca10 - Gauss	0.549	0.869	0.816	0.902	0.884

The Naive Bayes version performs much worse, with performance on alternative applications comparable to a random choice. This doesn't surprise us that much due to the fairly high correlation level between features, and also because we had a similar effect in the MVG Naive Bayes classifier.

Gaussian Mixture Model Tied Covariance Classifier					
	π(0.5), cfn(1), cfp(1)	π(0.8), cfn(1), cfp(1)	π(0.2), cfn(1), cfp(1)	π(0.5), cfn(10), cfp(1)	π(0.5), cfn(1), cfp(10)
Raw	**0.419**	0.766	0.704	0.927	0.842
Gauss	0.618	0.868	0.922	0.945	0.982
Pca10	0.423	0.774	0.710	0.937	0.840
Pca5	0.563	0.894	0.787	0.991	0.892
Pca10 - Gauss	0.710	0.922	0.957	0.961	0.997

Also the Tied Covariance model does not perform particularly well. This may be due to the multiple recomputation of the covariance matrix and the constraints applied on its eigenvalues which may have reduced the accuracy, and eliminated the benefit of having it computed over more data.





The Bayes plot from our best performing version of the model shows very calibrated scores on one side of the priors spectrum, and highly uncalibrated results on the other side.
Summing up, the Gaussian Mixture models classifier doesn't perform badly in general, but worse than a normal MVG. We won't consider it for our final classification task.

Final analysis with evaluation data

After verifying the performance of the classifiers and choosing their hyperparameters, we proceed to testing the model on our evaluation data, which has been left out of the analysis up to now.
We will test side by side our two best performing models up to now, the Support Vector Machine with the RBF Kernel and the Multivariate Gaussian Classifier with Tied Covariance Matrix.
Even though the latter has close performance to other models we tested, it is the one with the best score on the main application, and this is the reason why it has been chosen.

We will compare the minimum DCF values, which has been our evaluation metric up to now, with the actual DCF values of the models, to understand their value in a real-world application, where an actual threshold would be chosen.
After that we will employ Cross Validation again to try to choose a suitable threshold and make the final comparison between minimum DCF, actual DCF and actual DCF with a calibrated threshold.

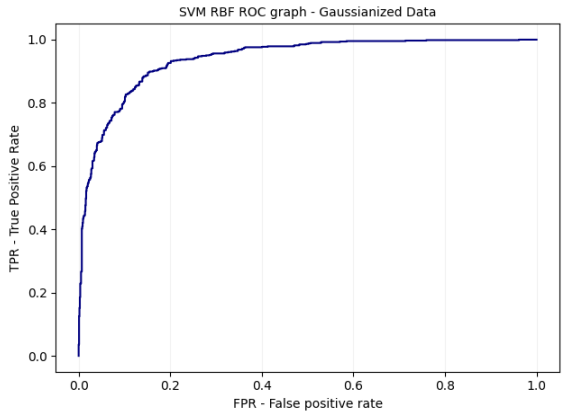
Final Evaluation on Classifiers		
	SVM RBF	Tied-Cov MVG
Minimum DCF	0.255	0.315
Actual DCF	0.328	0.337
Actual DCF - calibrated	0.279	0.351

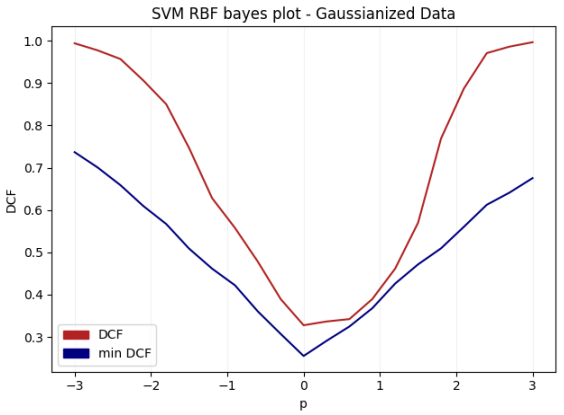
The SVM RBF confirms its results on the evaluation set with the lowest minimum DCF; it does have issues with threshold calibration as we can see from the significantly worse result of the actual DCF.

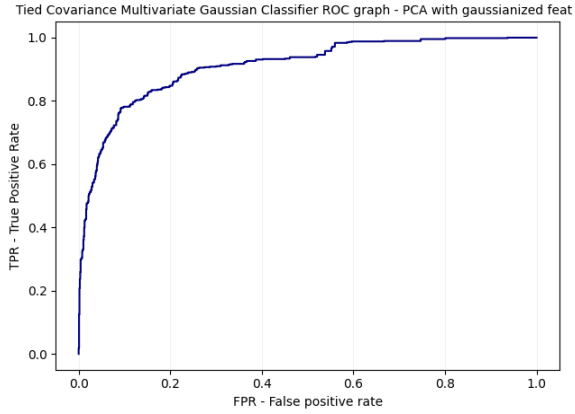
After the calibration procedure though, we achieve a very good DCF with the chosen threshold of -0.74, where the model obtains very good results, making it our model of choice in case we need to classify new samples.

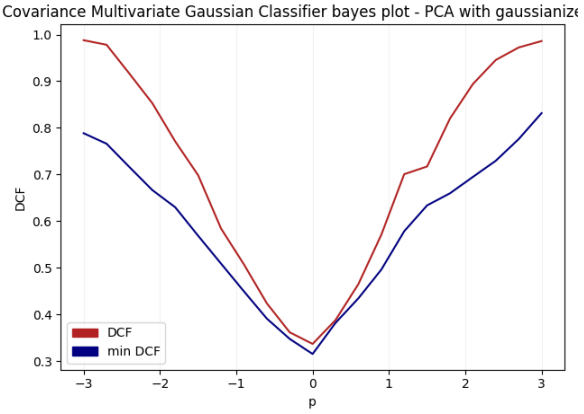
The Tied Covariance MVG performs slightly better than our previous tests in terms of minimum DCF, and the actual DCF isn't too far from the minimum value, showing that the scores are already not very uncalibrated.

Unfortunately the calibration process doesn't help, and makes the actual DCF score worse; this may be due to the distribution of the train samples in the split of the cross validation, and could possibly be improved by trying a different shuffle of the dataset.









From the graphs we can see the bad calibration of the scores of the SVM RBF around our application point, as we have seen from the actual DCF value. The ROC graphs is coherent with the final threshold chosen, as the TPR increases very quickly on the left side of the graph.

The Bayes plot of the Tied Covariance MVG shows that around our application point the scores are already fairly calibrated, as we have seen with the minimum and actual DCF scores being fairly similar from the start.

Concluding, our best approach confirms to be the Support Vector Machine with the kernel, where we achieve a DCF with calibrated score of 0.279 on our primary application.
The hyper-parameter choices we made during the training phase seem to be correct even with the evaluation data.
The results are similar between validation and evaluation set, and this isn't surprising due to the nature of the SVM, which has good performance already with a few samples.