Advanced Machine Learning Final Assignment

Emanuele Giuseppe Maria Maita 1000008891

1 Introduction

The following analysis is based on the "winequality-white" dataset (available at UCI repository[1]) provided by P. Cortez, A. Cerdeira, F. Almeida, T. Matos and J. Reis and featured on the article "Modeling wine preferences by data mining from physicochemical properties" in Decision Support Systems, Elsevier, 47(4) pag. 547-553, 2009[2])

The learning task concerns quality evaluation of wines, which is usually part of the certification process and can be used to improve production and to find premium brands. Wine certification is generally assessed by physicochemical and sensory tests, but the relationships between the two are complex and still not fully understood. The abovementioned paper presents a case study for modeling taste preferences based on physicochemical data that are easily available at the wine certification step, in particular using support vector machines and neural networks in the context of a regression task.

The dataset consists on the CVRVV[3] certification data of *vinho verde* white wines from Portugal, collected from May/2004 to February/2007. Each instance shows eleven physicochemical property of the wine and the quality evaluation, which is the median of a minimum of three sensory assessors (using blind tastes), which graded the wine in a scale that ranges from 0 (very bad) to 10 (excellent).

The aim of this work is to make prediction of the variable *quality* based on physicochemical properties. Section 2 deals with exploratory data analysis, based on univariate distributions, conditional distributions given the target and, after data normalization, principal component analysis. Section 3 is training; predicting *quality* is treated as a classification task, featuring the training (with hyperparameters tuning), evaluation and comparison of different classifiers using all features; next, the models will be trained on features extracted by linear discriminant analysis; subsequently, the task will be treated a a regression one, with the aim of exploiting the information given by the inherent order in *quality*. Once again into a classification framework, it follows an attempt to deal with the high class imbalance present in the data using SMOTE and a different metric for evaluation; then, the problem will be simplified, reducing the values of the target variables and training and evaluating again the models. The training section ends with a few unsupervised and semi-supervised observations. Finally, in the last section, the performance of the best model will be evaluated on unseen data.

2 Exploratory Data Analysis

2.1 Dataset Preview

The dataset contains 4898 instances, 11 real features (float numbers), which are *fixed acidity*, *volatile acidity*, *citric acid*, *residual sugar*, *chlorides*, *free sulfur dioxide*, *total sulfur dioxide*, *density*, *pH*, *sulphates* and *alcohol*, and an integer feature, *quality*. There are no null values.

[1] https://archive.ics.uci.edu/ml/datasets/wine+quality

[2] https://www.sciencedirect.com/science/article/abs/pii/S0167923609001377

[3] The CVRVV is an inter-professional organization with the goal of improving the quality and marketing of *vinho verde*.

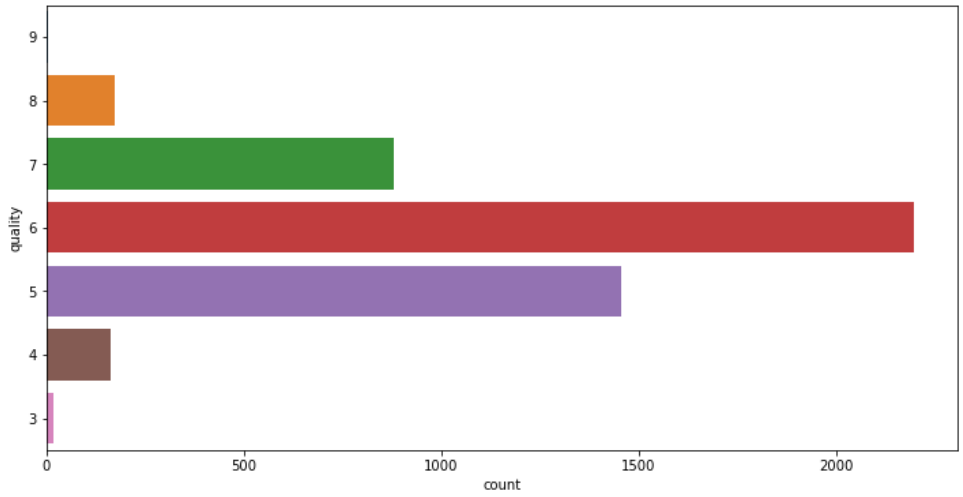
```
      fixed acidity  volatile acidity  citric acid  residual sugar  chlorides  \
0               7.0              0.27         0.36            20.7      0.045
1               6.3              0.30         0.34             1.6      0.049
2               8.1              0.28         0.40             6.9      0.050
3               7.2              0.23         0.32             8.5      0.058
4               7.2              0.23         0.32             8.5      0.058
           ...               ...          ...              ...        ...
4893            6.2              0.21         0.29             1.6      0.039
4894            6.6              0.32         0.36             8.0      0.047
4895            6.5              0.24         0.19             1.2      0.041
4896            5.5              0.29         0.30             1.1      0.022
4897            6.0              0.21         0.38             0.8      0.020
      free sulfur dioxide  total sulfur dioxide  density    pH  sulphates  alcohol  quality
0                    45.0                 170.0  1.00100  3.00       0.45      8.8        6
1                    14.0                 132.0  0.99400  3.30       0.49      9.5        6
2                    30.0                  97.0  0.99510  3.26       0.44     10.1        6
3                    47.0                 186.0  0.99560  3.19       0.40      9.9        6
4                    47.0                 186.0  0.99560  3.19       0.40      9.9        6
                    ...                   ...      ...   ...        ...      ...      ...
4893                 24.0                  92.0  0.99114  3.27       0.50     11.2        6
4894                 57.0                 168.0  0.99490  3.15       0.46      9.6        5
4895                 30.0                 111.0  0.99254  2.99       0.46      9.4        6
4896                 20.0                 110.0  0.98869  3.34       0.38     12.8        7
4897                 22.0                  98.0  0.98941  3.26       0.32     11.8        6
[4898 rows x 12 columns]
```

```
Data columns (total 12 columns):
 #   Column                Non-Null Count  Dtype
---  ------                --------------  -----
 0   fixed acidity         4898 non-null   float64
 1   volatile acidity      4898 non-null   float64
 2   citric acid           4898 non-null   float64
 3   residual sugar        4898 non-null   float64
 4   chlorides             4898 non-null   float64
 5   free sulfur dioxide   4898 non-null   float64
 6   total sulfur dioxide  4898 non-null   float64
 7   density               4898 non-null   float64
 8   pH                    4898 non-null   float64
 9   sulphates             4898 non-null   float64
 10  alcohol               4898 non-null   float64
 11  quality               4898 non-null   int64
dtypes: float64(11), int64(1)
```

2.2 Quality

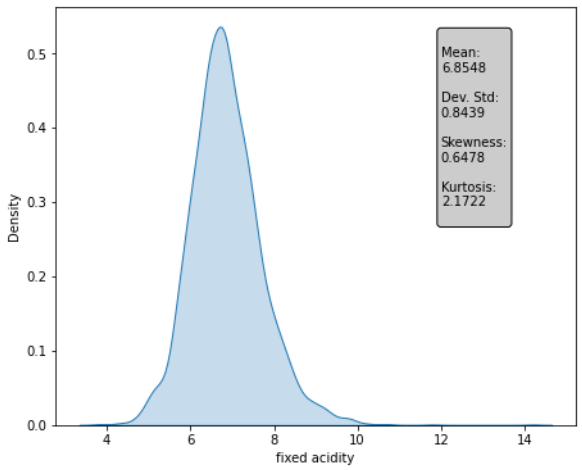
Quality is the target variable. The observations have a minimum value of 3 and a maximum of 9, with mode equal to 6. It shall be taken into account that the variable has an intrinsic order (information that is lost treating it as categorical in classification) and the dataset is highly unbalanced: in particular observations with quality 3,4,8,9 are quite fewer with respect to the modal class' ones.

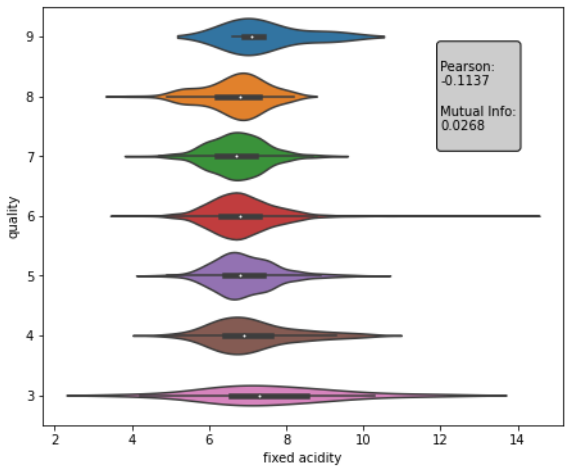
```
6    2198
5    1457
7     880
8     175
4     163
3      20
9       5
Name: quality, dtype: int64
```



2.3 *Fixed acidity*

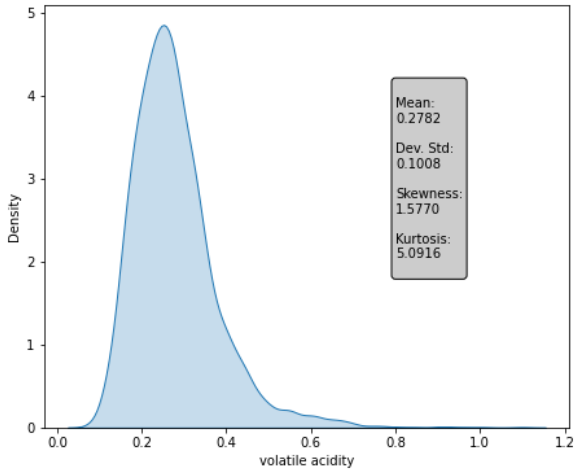
Fixed acidity has a mean value of about 6.9 and a standard deviation of about 0.84; it is right skewed and presents excess kurtosis, exhibiting non-normality. The violinplot shows, for different values of *quality*, how the conditional distribution changes, highlightning at the same time the median (white dot) and the interquartilic range (black rectangle) like a boxplot. It's possible to see that the modal class is responsable for the right tail in the distribution (for a lot of features, the quality value of 6 is responsible for the greater part of the outliers; this is in part due to the high difference in frequencies). It presents no meaningful correlation (Pearson linear coefficient) or mutual information (a more general form of dependence) with *quality*.

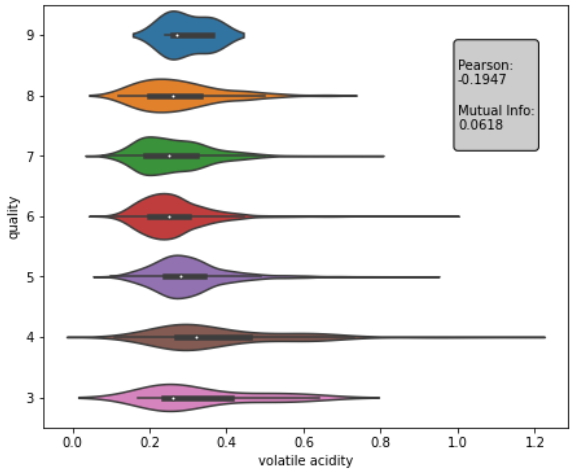




2.4 *Volatile acidity*

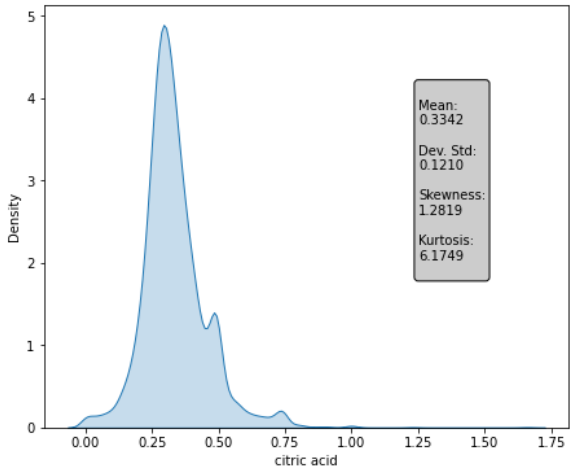
Volatile acidity has a mean value of 0.28 and standard deviation of 0.10; it's highly right skewed and leptokurtic, where classes from 4 to 6 are the most responsible for the right tail. The (negative) linear correlation is subtle; putting aside the tails and focusing on interquartilic ranges, it seems like *quality* oscillates around about 0.3 volatile acidity.

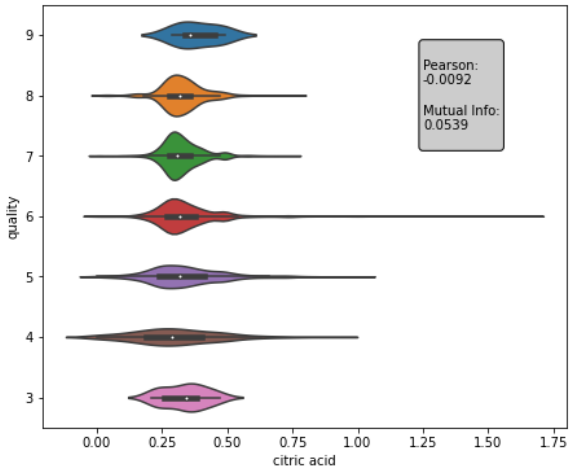




2.5 *Citric Acid*

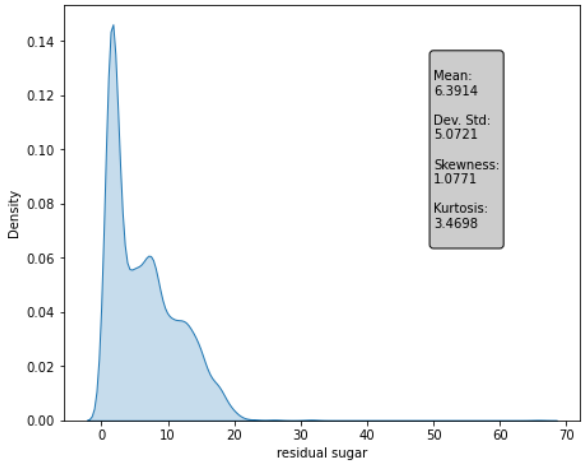
Citric Acid has a mean value of 0.33 and standard deviation 0.12; it's a multimodal distribution (where the peaks are not correlated with difference in quality), highly leptokurtic and right skewed (due to the modal class).

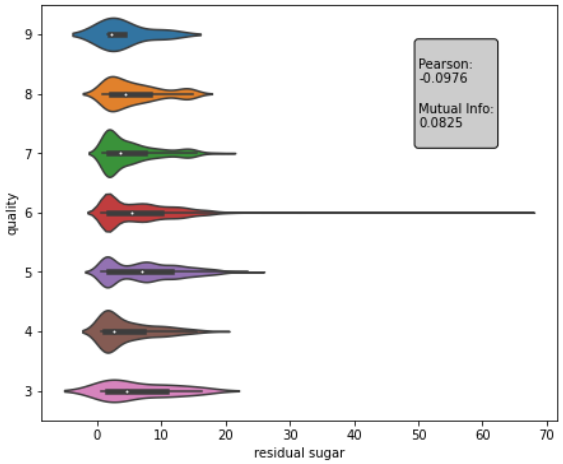




2.6 *Residual Sugar*

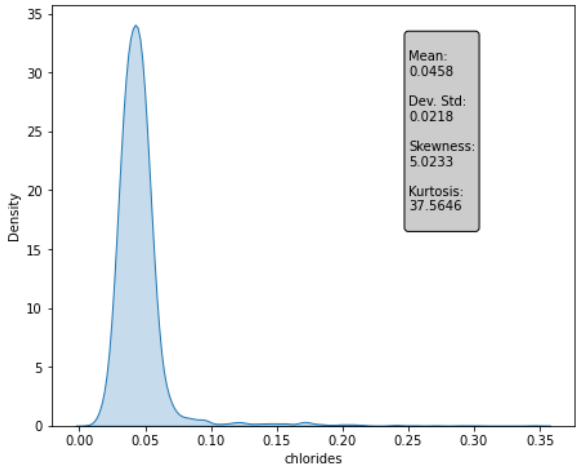
Residual sugar has mean of 6.39 and a relative high standard deviation of 5.07; it's a multimodal distribution (where multimodality is present in all classes), leptokurtic and right skewed; class 6 is responsible for the long tail of the distribution; the correlation with *quality* is subtle, but it seems like there are slightly higher values of residual sugar close to the central classes (5,6).

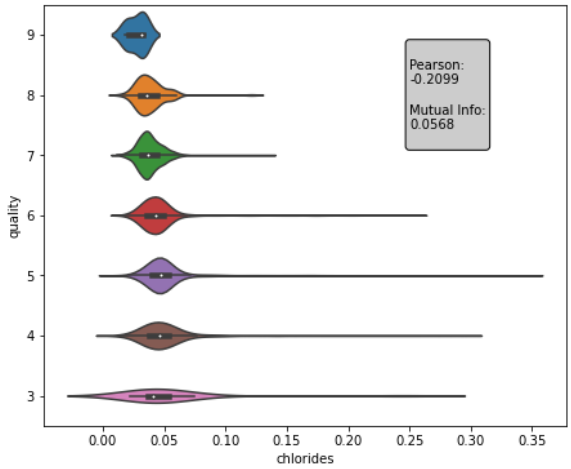




2.7 Chlorides

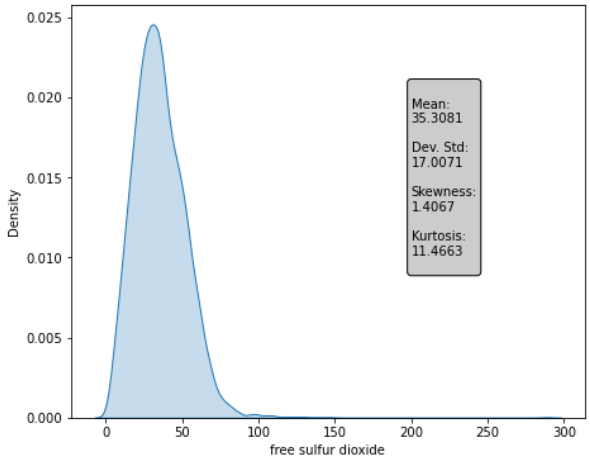
Chlorides has mean of 0,05 and a relative high standard deviation of 0.02; it presents a very long right tail and very high value of excess kurtosis. There is a negative linear correlation of about 21% with the target, with lower classes showing higher values (mainly outliers).

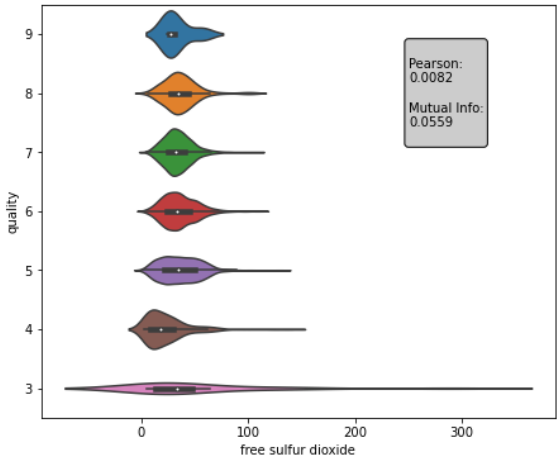




2.8 Free sulfur dioxide

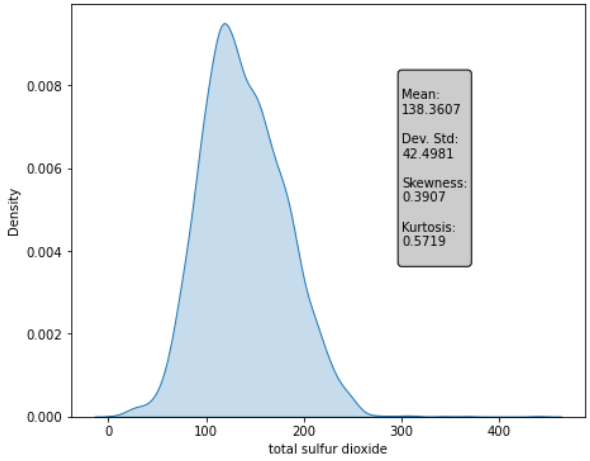
Free sulfur dioxide has mean of about 35 and a standard deviation of 17; it is right skewed and leptokurtic. Class 4 distribution seems to gravitate on slightly smaller values than other classes, but, generally, there is very low correlation with *quality*.

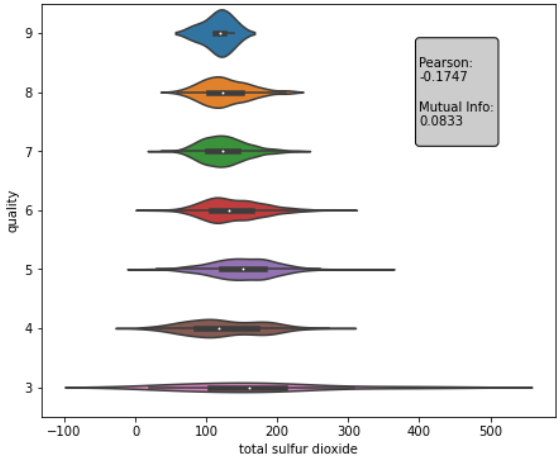




2.9 Total sulfur dioxide

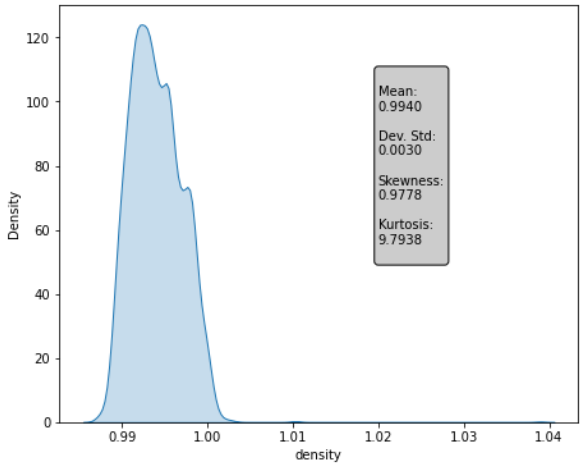
Total sulfur dioxide has mean of about 138 and a standard deviation of 42; this feature is on a very different scale with respect to the others (together with *free sulfur dioxide*), and it justifies the use of a normalization procedure to apply scale-sensitive algorithms. It presents some right skewness and excess kurtosis; it shows a negative linear correlation of 17% with *quality* and, while is more spread out for lower classes, it's more concentrated towards the mean value for higher classes.

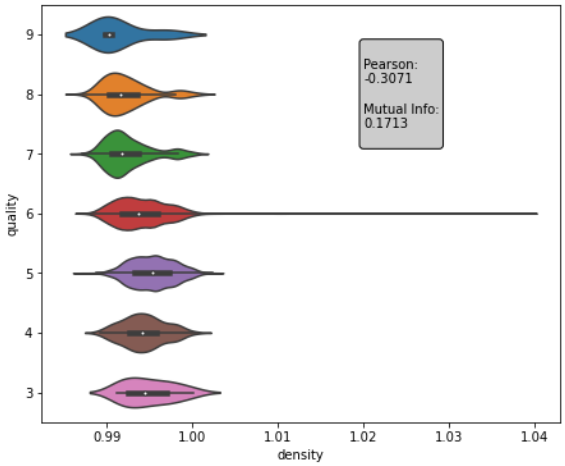




2.10 Density

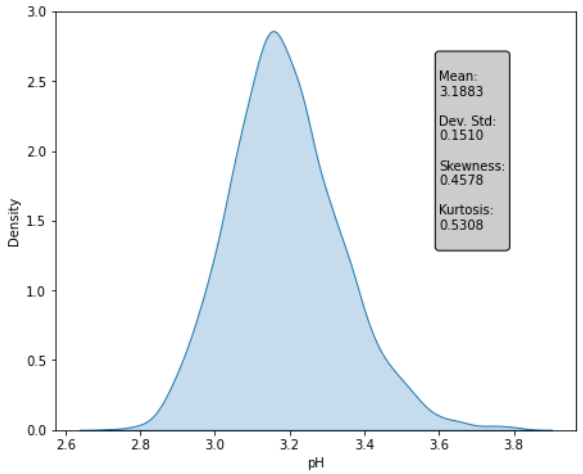
Density has mean of about 0.99 and a relative low standard deviation of 0.003; it has a multimodal distribution with a long right tail due to the modal class. It presents a moderate negative linear correlation of 30% (and a relative high value of mutual information): indeed we can see that lower density is correlated with higher quality.

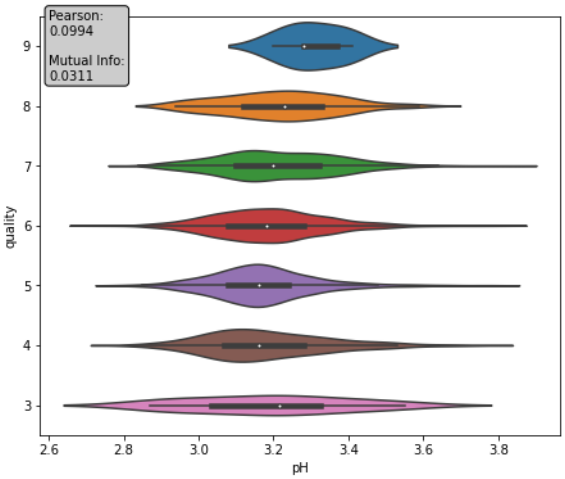




2.11 pH

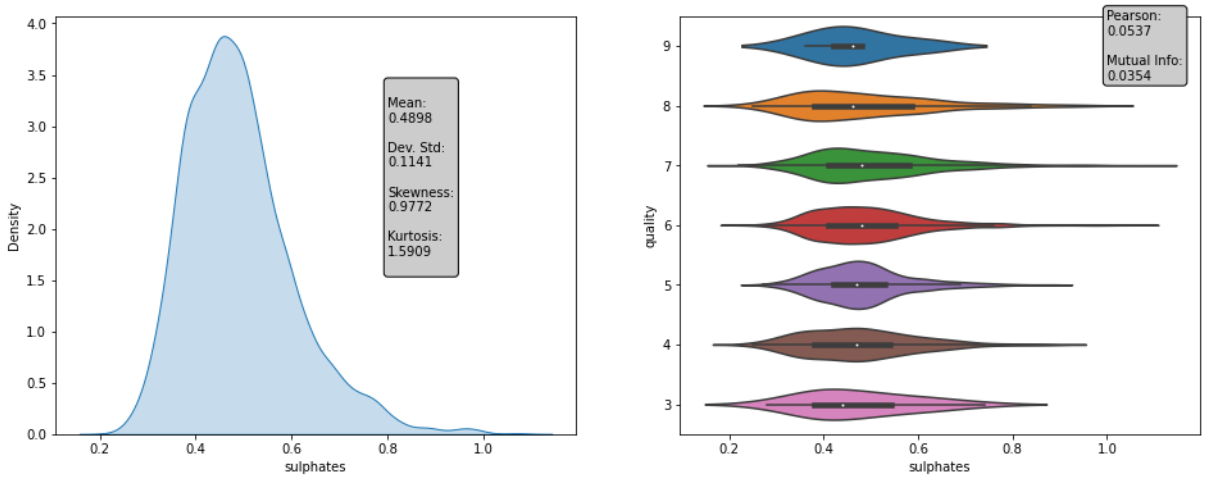
PH has mean of about 3 and a relative low standard deviation of 0.15; it's the most symmetric distribution, with, nonetheless, some departure from normality (right skewness, excess kurtosis). Looking at interquartilic ranges, higher classes of quality are more concentrated around slightly higher values of *pH*.





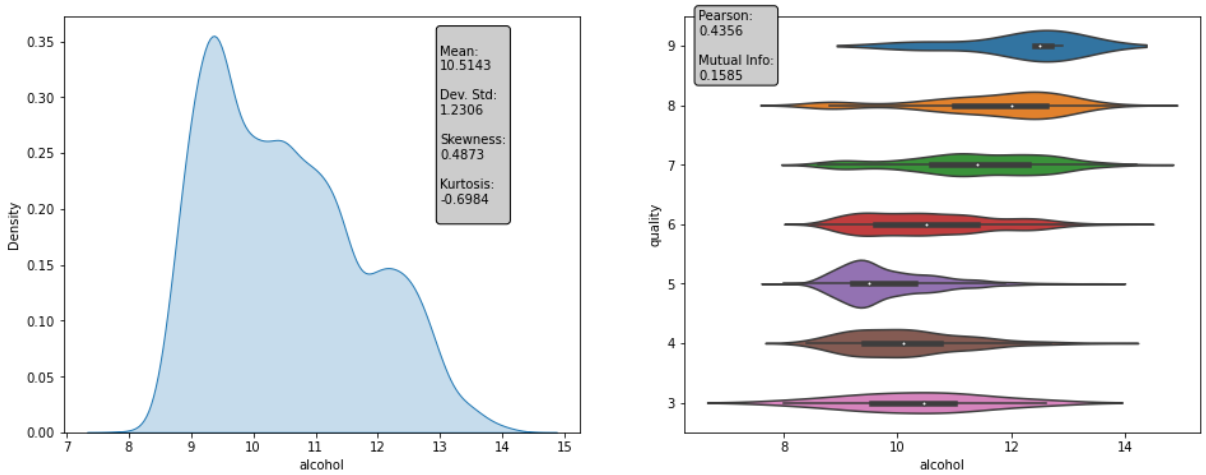
2.12 Sulphates

Sulphates has mean of about 0.48 and a standard deviation of 0.11; its distribution is right skewed and leptokurtic. It presents low correlation with *quality*.



2.13 Alcohol

Alcohol has mean of about 10.50 and a standard deviation of 1.2; its distribution is multimodal, right skewed and platykurtic. It seems to be the feature most correlated with *quality*, with a linear correlation coefficient of 0.43; class 3 to 5 concentrate on lower and lower values; then, there is a relatively dramatic increase in alcohol with class 6 up to 9.



2.14 Normalization and Principal Component Analysis

Before the training phase, normalization is required by a lot of different machine learning algorithms (such as distance-based methods like K-nearest neighbors and SVM), which do not perform well when different features have different scales; moreover, principal component analysis will be performed next, another procedure which is highly sensible to scaling. As a consequence, the dataset will be scaled into $[0, 1]$ range using the min-max scaler, independent to any assumption on the data distribution (given the diffused non-normality, standardization is inadequate):

$$x_{new} = \frac{x - x_{min}}{x_{max} - x_{min}}$$

for each feature (x) in the dataset.

	fixed acidity	volatile acidity	citric acid	residual sugar	chlorides	\
0	0.307692	0.186275	0.216867	0.308282	0.106825	
1	0.240385	0.215686	0.204819	0.015337	0.118694	
2	0.413462	0.196078	0.240964	0.096626	0.121662	
3	0.326923	0.147059	0.192771	0.121166	0.145401	
4	0.326923	0.147059	0.192771	0.121166	0.145401	
	...	...	...	...	...	
4893	0.230769	0.127451	0.174699	0.015337	0.089021	
4894	0.269231	0.235294	0.216867	0.113497	0.112760	
4895	0.259615	0.156863	0.114458	0.009202	0.094955	
4896	0.163462	0.205882	0.180723	0.007669	0.038576	
4897	0.211538	0.127451	0.228916	0.003067	0.032641	

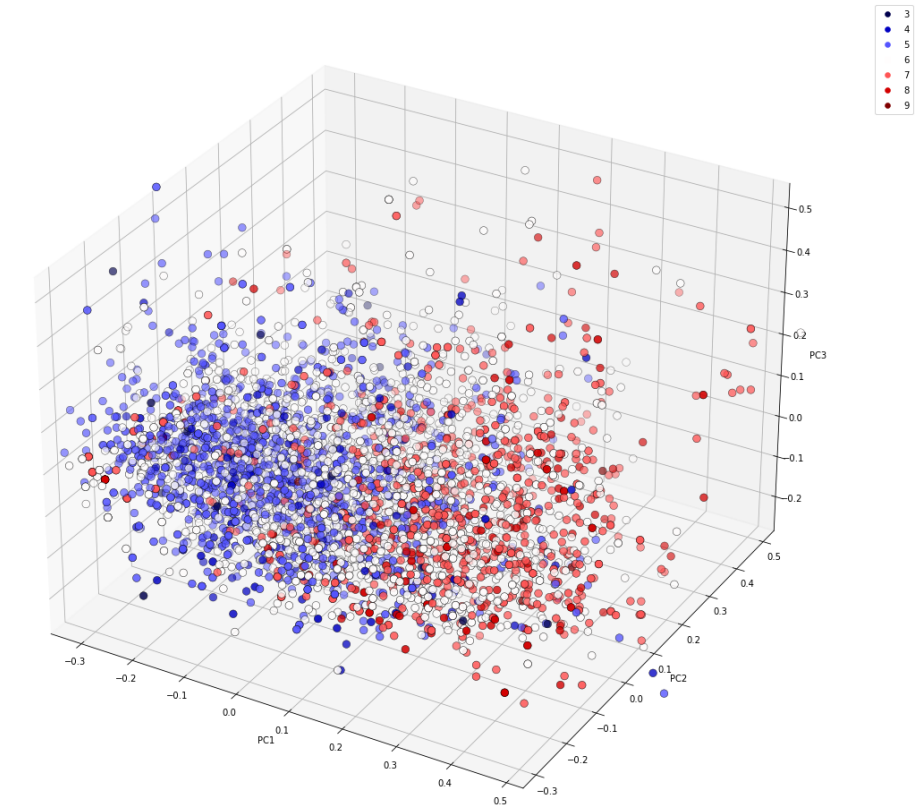
	free sulfur dioxide	total sulfur dioxide	density	pH	sulphates	alcohol
0	0.149826	0.373550	0.267785	0.254545	0.267442	0.129032
1	0.041812	0.285383	0.132832	0.527273	0.313953	0.241935
2	0.097561	0.204176	0.154039	0.490909	0.255814	0.338710
3	0.156794	0.410673	0.163678	0.427273	0.209302	0.306452
4	0.156794	0.410673	0.163678	0.427273	0.209302	0.306452
	...	...	...	...	...	...
4893	0.076655	0.192575	0.077694	0.500000	0.325581	0.516129
4894	0.191638	0.368910	0.150183	0.390909	0.279070	0.258065
4895	0.097561	0.236659	0.104685	0.245455	0.279070	0.225806
4896	0.062718	0.234339	0.030461	0.563636	0.186047	0.774194
4897	0.069686	0.206497	0.044342	0.490909	0.116279	0.612903

[4898 rows x 11 columns]

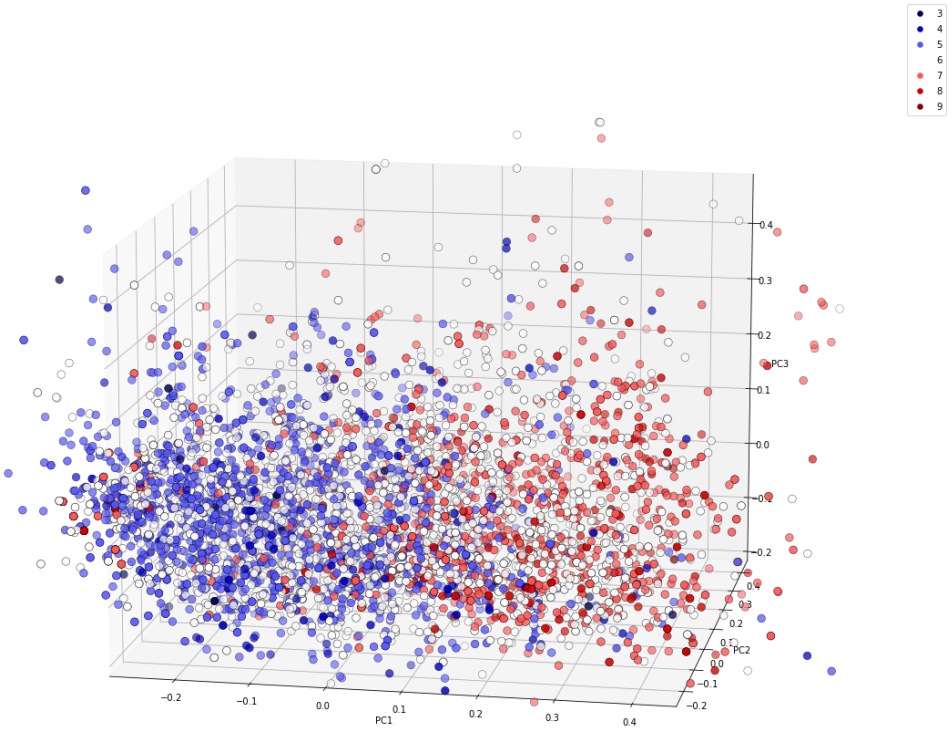
The first principal component is that linear combination of the features (it's an unsupervised procedure, so the target is excluded) which maximize the amount of variability (i.e. of information) present in the data. The subsequent principal components are the orthogonal linear combinations which maximize the amount of variance not contained in prior components. If one computes the cumulative proportion of variance explained by the principal components, it can be assessed if there can be a meaningful low-dimensional representation of the data: in this case, the first 3 principal component show just under 70% of the information in the data.

	PC cumulative variance explained
PC1	0.381660
PC2	0.560123
PC3	0.689076
PC4	0.775004
PC5	0.843432
PC6	0.889391
PC7	0.928632
PC8	0.961846
PC9	0.986172
PC10	0.999257
PC11	1.000000

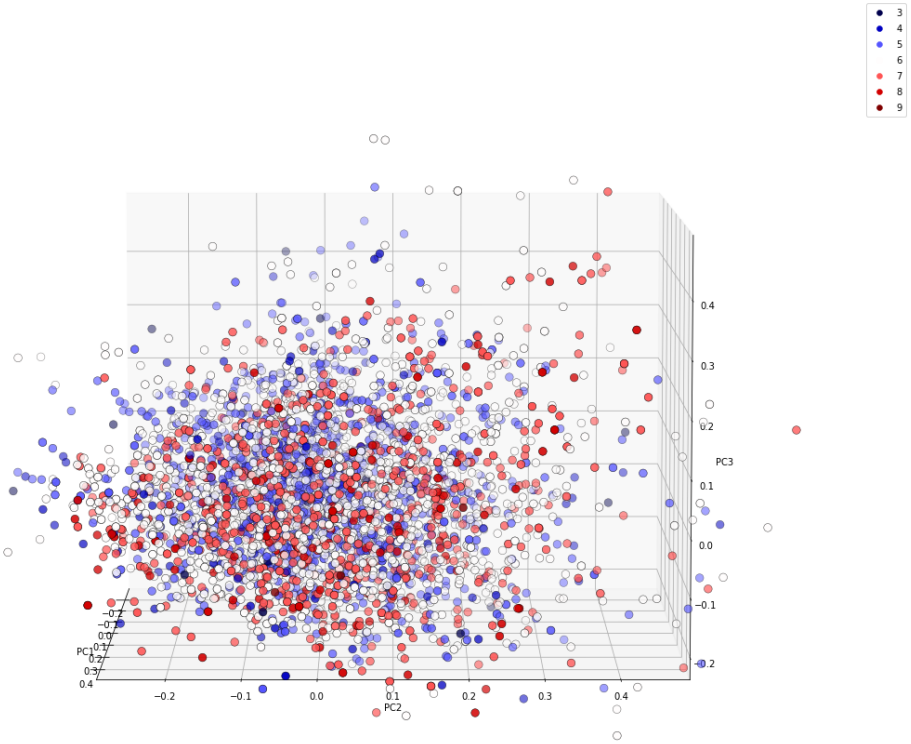
Based on that, a three dimensional representation of the data (i.e. using the first three principal component) can be useful to have a glimpse of how the data is distributed. In the following scatterplot, colors relate to the quality classes: to enhance the different distribution in principal component space of lower and higher quality values, a divergent color palette is used, where lower the class, more blue-ish the color, greater the class, more red-ish the color; the middle (and modal) class 6 appears as white (and it's basically uniformly distributed in space).



The first principal component appears to be the most relevant to distinguish lower from higher classes:



while the other two principal components are not so useful:



From the observation that the first principal component appears to be useful in discriminating *quality*, it could be interesting to compute the linear correlations with all features, which determines the more important variables in its construction: e.g. alcohol and density are the most significative ones, result that confirm their relevancy as it has been already capture by association measures already computed (here summarized).

	correlation with PC1	correlation with quality	M.I. with quality
fixed acidity	-0.199902	-0.113663	0.026821
volatile acidity	0.049065	-0.194723	0.061779
citric acid	-0.137593	-0.009209	0.053899
residual sugar	-0.564916	-0.097577	0.082454
chlorides	-0.383471	-0.209934	0.056820
free sulfur dioxide	-0.344486	0.008158	0.055889
total sulfur dioxide	-0.568606	-0.174737	0.083284
density	-0.849399	-0.307123	0.171322
pH	0.220472	0.099427	0.031108
sulphates	-0.034353	0.053678	0.035425
alcohol	0.977175	0.435575	0.158455

3 Training

3.1 Training/test split

The full dataset is now split into two sets, stratifying according to the target variable (i.e. each set contains approximately the same proportion of instances per class), based on a 80/20 proportion.

The first set (`X_train`, `y_train`) will be given as input, during training phase, to a (stratified) 3-fold cross validation procedure, where each model will be tested three times: on each fold, after being trained on the remaining two; the accuracy of the model will be averaged among the three. With cross validation, it's possible to obtain a less noisy

accuracy evaluation (it does not depend on a singular choice of a validation set) and to train the final model on the full (`X_train`, `y_train`) set.

The second set (`X_test`, `y_test`) will be used to obtain an independent evaluation of the accuracy of the best model (i.e. with data not used for hyperparameter tuning).

3.2 Extremely Randomized Trees

A decision tree is a non-parametric supervised learning method that use a set of binary rules, the ones which minimize some impurity measure, to calculate a target value. The predictor space is segmented in a number of simple regions, that can be summarized using a tree representation.

Single (and unpruned) decision trees are prone to overfitting and exhibit high variance; better results are generally obtained using an ensemble method, like extremely randomized trees: with the purpose of reducing variance and overfitting, it averages the responses of multiple decision trees fitted on a bootstraped sample or the entire dataset, where the number of features considered for each split can be reduced (i.e. a subset of predefined size is randomly generated at every split) and each split is the best of randomly-generated thresholds.

According to 3-fold cross-validation, the best extremely randomized trees ensemble among those defined by the following parameter grid:

```
parameters = {
    'n_estimators':[2,5,10,20,50,100,200,500,1000],
    'bootstrap':[True, False],
    'criterion':['gini','entropy'],
    'max_features':[2,'sqrt', None],
    'max_depth':[5, 10, None],
    'class_weight':[None,'balanced','balanced_subsample']
}
```

is the one computed using cross-entropy as impurity criterion, 500 tree estimators, with 2 features at a time to choose from during node splitting and class weights inversely proportional to class frequencies, and without bootstrap or limitation in terms of depth. The accuracy is 0.65799.

A relevant information which can be retrieved after fitting the model is given by the (normalized) total reduction of impurity brought by each feature, the "feature importances": greater importance means greater impact in determining *quality*. Not surprisingly (considering the other measures, here once again reported), the most important variable is *alcohol*; an interesting result is given by the second feature in this ranking, *free sulfur dioxide*, which relevancy has not been spotted by the other measures; moreover, *density*, the most correlated variable after alcohol, is only in sixth place and chlorides, with a negative linear correlation of 21%, is at the bottom.

	feature importance	corr. with quality	M.I. with quality	corr. with PC1
alcohol	0.137350	0.435575	0.158455	0.977175
free sulfur dioxide	0.113238	0.008158	0.055889	-0.344486
total sulfur dioxide	0.099355	-0.174737	0.083284	-0.568606
fixed acidity	0.088717	-0.113663	0.026821	-0.199902
volatile acidity	0.087425	-0.194723	0.061779	0.049065
density	0.087254	-0.307123	0.171322	-0.849399
pH	0.083167	0.099427	0.031108	0.220472
residual sugar	0.079930	-0.097577	0.082454	-0.564916
sulphates	0.077656	0.053678	0.035425	-0.034353
citric acid	0.075730	-0.009209	0.053899	-0.137593
chlorides	0.070178	-0.209934	0.056820	-0.383471

Feature importance can be used to discharge recursively the most irrelevant variables and re-fit the model in a dataset with less dimensions. At each step, it's possible to compute the k-fold cross-validation accuracy and then make a comparison: the step with higher cv accuracy suggest the optimal number of features. Recursive feature selection performed using the best ensemble model with 3-fold cross validation where at each step the single worst variable is removed, suggests to not discharge any of the feature in the dataset. As a consequence of this result, all models will be trained using the entire dataset, without any form of variable selection (aside for feature extraction in *3.6*).

3.3 K-Nearest Neighbors

K-Nearest-Neighbors is an instance-based learning algorithm which assigns any new data to the class which is the most common in its neighborhood (defined by the training set).

According to 3-fold cross-validation, the best K-nearest neighbors model among those defined by the following parameter grid:

```
parameters = {
    'n_neighbors':[1,2,3,4,5,6,7,8,9,10,20,40,50,80,100,120,150,180,200],
    'weights':['uniform', 'distance'],
    'p':[1,2]
}
```

uses Manhattan distance and 200 neighbors, where each neighbor influence is inversely related to its distance. The accuracy score is 0.63808.

3.4 Support Vector Machines

Support vector machines search for the optimal separating hyperplane, i.e. the one which separates the classes being the farthest from the training observations (as measured by the margin) and thus the most stable under perturbations of the inputs; the model allows for some observations to be on the incorrect side of the margin or in the incorrect side of the hyperplane (defining a "soft" margin) based on a cost parameter (C). Moreover, with application of some kernel function (different from the linear one), the features are transformed before fitting, in a representation where (hopefully) the observations can be (linearly) separated more easily (increasing accuracy). According to 3-fold cross-validation, the best SVM model among those defined by the following parameter grids[4]:

```
parameters = [
    {
     'clf': (SVC(),),
     'clf__C': [0.001, 0.01, 0.1, 1,10],
     'clf__kernel': ['linear'],
     'clf__class_weight':[None,'balanced']
     },
    {
     'clf': (SVC(),),
     'clf__C': [0.001, 0.01, 0.1, 1,10],
     'clf__kernel': ['poly'],
     'clf__degree': [2,3,5],
     'clf__class_weight':[None,'balanced']
     },
    {
     'clf': (SVC(),),
```

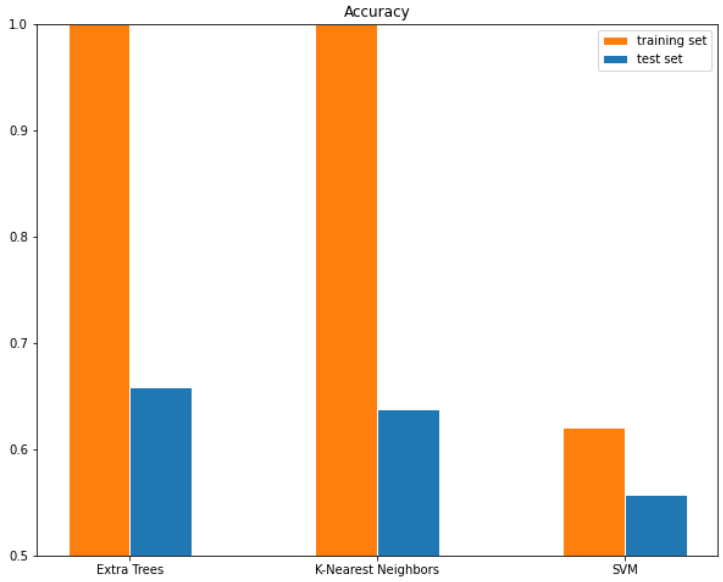
[4]without using different parameter grids, the computation would include duplicate models.

```
        'clf__C': [0.001, 0.01, 0.1, 1,10],
        'clf__kernel': ['rbf'],
        'clf__gamma': [0.5,1,2,'scale','auto'],
        'clf__class_weight':[None,'balanced']
        }
]
```

uses a radial basis function kernel, with gamma equal to $1/[n\_features \cdot variance(\texttt{X\_train})]$, i.e. the "scale" value, and cost parameter equal to 10. The accuracy is 0.55692.

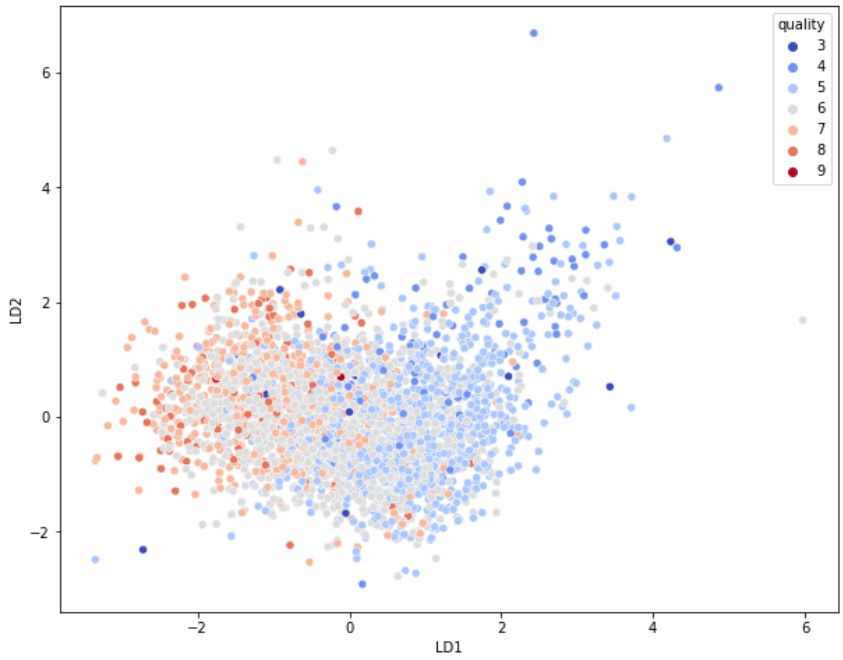
3.5 Model Comparison

Among those trained, the best classifier on this dataset according to cross validation accuracy is the extremely randomized trees ensemble (about 66%).



3.6 Training in LDA space

Linear discriminant analysis finds those K dimensions (where K is the number of classes) which best linearly separate the data; the following is a representation of the data in the space spanned by the first two LDA components (color per class similar to the PCA representation):



It's possible to perform LDA as a feature extraction step prior to training, so that each model can be trained on fewer variables (K or less, where K in this case is 6). Using the same models and parameters grids in *3.2*, *3.3* and *3.4* and adding a parameter for lda components (2 or 6):

```
parameters = [{
    'lda__n_components':[2,6],
    'clf' : (ExtraTreesClassifier(random_state=0),),
    'clf__n_estimators':[2,5,10,20,50,100,200,500,1000],
    'clf__bootstrap':[True, False],
    'clf__criterion':['gini','entropy'],
    'clf__max_features':[2,'sqrt', None],
    'clf__max_depth':[5, 10, None],
    'clf__class_weight':[None,'balanced','balanced_subsample']
},{
    'lda__n_components':[2,6],
    'clf' : (KNeighborsClassifier(),),
    'clf__n_neighbors':[1,2,3,4,5,6,7,8,9,10,20,40,50,80,100,120,150,180,200],
    'clf__weights':['uniform', 'distance'],
    'clf__p':[1,2]
},{
   'lda__n_components':[2,6],
   'clf': (SVC(),),
   'clf__C': [0.001, 0.01, 0.1, 1, 10],
   'clf__kernel': ['linear'],
   'clf__class_weight':[None,'balanced']
},{
   'lda__n_components':[2,6],
   'clf': (SVC(),),
   'clf__C': [0.001, 0.01, 0.1, 1, 10],
   'clf__kernel': ['poly'],
   'clf__degree': [2,3,5],
   'clf__class_weight':[None,'balanced']
},{
   'lda__n_components':[2,6],
   'clf': (SVC(),),
   'clf__C': [0.001, 0.01, 0.1, 1, 10],
   'clf__kernel': ['rbf'],
   'clf__gamma': [0.5,1,2,'scale','auto'],
   'clf__class_weight':[None,'balanced']
}]
```

according to 3-fold cross-validation, the best model in 6-dimensional LDA space is the extra trees ensemble with cross-entropy criterion, balanced subsample class weight[5], 2 as max_features, 500 as number of estimators, with the use of bootstrap and maximal depth. It reaches an accuracy of 0.64267. Using only the first two components of LDA for training, the best model is the K-nearest neighbor classifier with K equal to 180, using Manhattan distance and weighted neighbors, with an accuracy of just about 0.03 less than the best model on the original 11-dimensional space: 0.63119.

3.7 Regression

Quality has an inherent order, an information which, so far, has been ignored. In the following, the target feature will be considered as a quantitative variable and, as such, modeled through regression: the regression counterpart of the algorithms used for classification will be trained and cross-validated using the same parameter space (except for the ensemble criterion, which will be mean square error).

```
parameters = [{
    'reg' : (ExtraTreesRegressor(random_state=0),),
    'reg__n_estimators':[2,5,10,20,50,100,200,500,1000],
    'reg__bootstrap':[True, False],
    'reg__max_features':[2,'sqrt', None],
    'reg__max_depth':[5, 10, None]
},{
    'reg' : (KNeighborsRegressor(),),
    'reg__n_neighbors':[1,2,3,4,5,6,7,8,9,10,20,40,50,80,100,120,150,180,200],
    'reg__weights':['uniform', 'distance'],
    'reg__p':[1,2]
},{
   'reg': (SVR(),),
   'reg__C': [0.001, 0.01, 0.1, 1, 10],
   'reg__kernel': ['linear']
},{
   'reg': (SVR(),),
   'reg__C': [0.001, 0.01, 0.1, 1, 10],
   'reg__kernel': ['poly'],
   'reg__degree': [2,3,5]
},{
   'reg': (SVR(),),
   'reg__C': [0.001, 0.01, 0.1, 1, 10],
   'reg__kernel': ['rbf'],
   'reg__gamma': [0.5,1,2,'scale','auto']
}]
```

To compare the best regression model with the classifiers' results, prediction values (which are reals) will be first rounded to the closest integer and then cross-validated with the usual accuracy metric. The best regressor model is the extra tree ensemble with 1000 tree estimators, 2 as max_features and no bootstrap. It is slightly different to the best classifier (more estimators and the MSE impurity) but the accuracy is approximately the same, 0.65646. The root mean squared error is equal to 0.62658 and the R^2 score (proportion of variance of the target explained by the features) is 49.84% (both these measures are cross-validated too). On the basis of this result, it can be stated that the inherent order in *quality* make no particular impact on the predictive power of the model.

[5]same as "balanced" except that weights are computed based on the bootstrap sample for every tree grown.

3.8 SMOTE and balanced accuracy

As mentioned, the dataset is quite imbalanced: extreme classes 3,4,8,9 have low counts relative to the central classes. In case of imbalanced classification, there are too few examples for the model to learn the decision boundaries of minority classes effectively. One way to deal with this problem is with a synthetic minority oversampling technique (SMOTE); specifically, the variant that will be applied here is a 1-nearest neighbors BorderlineSMOTE[6] approach to generate new samples: given a random example from the minority class and its nearest neighbors of the same class (default kind = "borderline-1") which are on the border (defined, as default, by the fact that at least half of the 10 nearest neighbors are of the same class, but not all), a synthetic example is generated as a randomly selected point between the hyperplane connecting the two examples in feature space.

Oversampling is considered as an hyperparameter to tune, with alternatives: no oversampling ({}), moderated oversampling (in particular the minority classes 3,4,8,9 will be augmented to 300 counts: {3:300, 4:300, 8:300, 9:300}), and full oversampling ({None}, i.e. all classes will have the same frequency, equal to the one of the majority class); it will be applied on each training set generated by cross validation, while every test set won't be modified.

Using the standard accuracy metric, any classifier is prone to learn well majority classes' boundaries, policy that generally bring more correctly classified observations, but not minority classes' boundaries (which are more difficult to learn and would bring relatively less classification counts). So, in the following, cross validation grid search will be applied using the balanced accuracy metric, i.e. where each sample is weighted according to the inverse prevalence of its true class.

```
parameters = [{
    'sampling__sampling_strategy':[{},{8:300, 4:300, 3:300, 9:300}, 'auto'],
    'clf' : (ExtraTreesClassifier(random_state=0),),
    'clf__n_estimators':[2,5,10,20,50,100,200,500,1000],
    'clf__bootstrap':[True, False],
    'clf__criterion':['gini','entropy'],
    'clf__max_features':[2,'sqrt', None],
    'clf__max_depth':[5, 10, None],
    'clf__class_weight':[None,'balanced','balanced_subsample']
},{
    'sampling__sampling_strategy':[{},{8:300, 4:300, 3:300, 9:300}, 'auto'],
    'clf' : (KNeighborsClassifier(),),
    'clf__n_neighbors':[1,2,3,4,5,6,7,8,9,10,20,40,50,80,100,120,150,180,200],
    'clf__weights':['uniform', 'distance'],
    'clf__p':[1,2]
},{
   'sampling__sampling_strategy':[{},{8:300, 4:300, 3:300, 9:300}, 'auto'],
   'clf': (SVC(),),
   'clf__C': [0.001, 0.01, 0.1, 1, 10],
   'clf__kernel': ['linear'],
   'clf__class_weight':[None,'balanced']
},{
   'sampling__sampling_strategy':[{},{8:300, 4:300, 3:300, 9:300}, 'auto'],
   'clf': (SVC(),),
   'clf__C': [0.001, 0.01, 0.1, 1, 10],
   'clf__kernel': ['poly'],
   'clf__degree': [2,3,5],
   'clf__class_weight':[None,'balanced']
},{
   'sampling__sampling_strategy':[{},{8:300, 4:300, 3:300, 9:300}, 'auto'],
```

[6] https://imbalanced-learn.org/stable/over_sampling.html

```
    'clf': (SVC(),),
    'clf__C': [0.001, 0.01, 0.1, 1, 10],
    'clf__kernel': ['rbf'],
    'clf__gamma': [0.5,1,2,'scale','auto'],
    'clf__class_weight':[None,'balanced']
}]
```

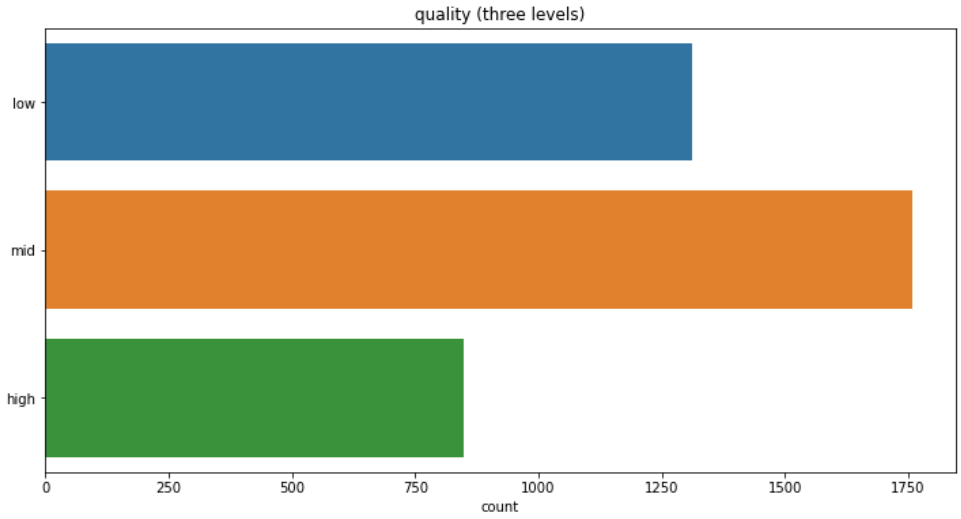
The best model according to 3-fold cross validated balanced accuracy is the extremely randomized trees ensemble that uses maximal oversampling with cross-entropy impurity, 500 tree estimators, max depth equal to 5, square root of the number of feature to choose from per split, weighted classes, no bootstrap. The score obtained is 0.44296, while the standard accuracy is 0.33384. To make a comparison, the model previously obtained, i.e. by optimizing according to the standard accuracy, have a balanced accuracy equal to 0.347675, while the standard accuracy is 0.657989.

	accuracy	balanced accuracy	macro f1	weighted f1
classifier	0.657989	0.347675	0.390107	0.642508
balanced classifier	0.333844	0.442956	0.237924	0.310074

Increasing the accuracy for minority classes is too costly: optimizing balanced accuracy brings an increase of 0.10 of this score with respect to the standard classifier, but with a tremendous decrease in term of global accuracy of 0.33 (about 50% less); moreover, the f1 score, which is the average of f1 scores of single classes, i.e. a metric which give same weight to precision and recall, decreases substantially, not only when weighted (where majority classes have more relevancy), but even when all classes contribute in the same way (macro f1).

3.9 Low/mid/high target classification

To boost model performance at the cost of losing information, one could extensively simplify the problem by consider only three classes of quality: low, mid and high. This could be done, dealing at the same time with class imbalance, by considering class 3,4 and 5 as low, 6 as mid, and 7,8 and 9 as high values.



The models and parameter grid will be the same as the ones used in *3.2*, *3.3* and *3.4*. The best model according to 3-fold cross validated accuracy is once again the trees

ensemble; in particular, it uses the Gini criterion, with the square root of the number of features during splitting, 200 as number of tree estimators, no class weighting, no limits on depth. Nevertheless, despite the costly simplification of the problem, the increase in performance is only about 0.04: the accuracy is 0.70010.

3.10 Unsupervised and Semi-Supervised learning

It could be interesting to make a comparison between the *quality* classes in training data and the latent classes that can be found using an unsupervised method, in search for clusters in the data. Considering the combinations of linkages "single", "complete", "average", "Ward" with affinity metrics "euclidean" and "manhattan"[7] for agglomerative hierarchical clustering and the KMeans algorithm, the models which clusters are more similar to the true labels according to the adjusted Rand index are:

- hierarchical clustering with complete linkage and manhattan affinity metric, with Rand index equal to 0.04786, for the original labeling;
- Kmeans, with Rand index equal to 0.07632, for the high/mid/low labeling.

Let's imagine now that only just a part of the (training) data was labeled at the beginning: e.g. the 20%[8]. Considering the two graph-based semi-supervised algorithm label propagation and label spreading (where the second uses a more noise-robust graph structure than the first) with the following parameters:

- kernel = ["knn", "rbf"]; these project data into alternate dimensional spaces.
  - n_neighbors for KNN kernel: [60, 70, 90, 100, 150];
  - gamma values for RBF kernel: [0.1, 0.2, 0.5, 1, 2, 5, 10, 20, 50];
- for LabelSpreading, alpha = [0.1, 0.2, 0.3, 0.5, 0.7]; this defines the maximum amount of labels which weight can be changed by the algorithm during the computation of the new labels.

the best ones are:

- LabelSpreading with RBF kernel, gamma value equal to 50 and alpha equal to 0.1, with an accuracy of 0.53270, for the original problem.
- LabelSpreading with RBF kernel, gamma equal to 50 and alpha equal to 0.5, with an accuracy of 0.57065, for the simplified three-valued problem.

4 Test and conclusion

Considering that optimizing balanced accuracy is too costly in terms of standard accuracy (*3.8*), and that a simplified account of quality is not fruitful enough given the high loss of information (*3.9*), the best model remains the one identified in *3.2*: the extremely randomized trees ensemble with cross-entropy as impurity criterion, 500 tree estimators, with

[7] with the exception of Ward-manhattan distance, not defined.

[8] It's not possible to augment the unlabeled data, considering that class 9 has only 5 observations.

2 features at a time to choose from during node splitting, without bootstrap or limitation in terms of depth, and with class weights inversely proportional to class frequencies. This is now tested using new, unseen data (`X_test`), for which quality labels will be predicted:

```
              precision    recall  f1-score   support

           3       0.00      0.00      0.00         4
           4       1.00      0.21      0.35        33
           5       0.75      0.69      0.72       291
           6       0.64      0.84      0.73       440
           7       0.76      0.44      0.56       176
           8       0.85      0.49      0.62        35
           9       0.00      0.00      0.00         1

    accuracy                           0.69       980
   macro avg       0.57      0.38      0.42       980
weighted avg       0.71      0.69      0.67       980


[[  0   0   2   2   0   0   0]
 [  0   7  15  11   0   0   0]
 [  0   0 201  89   1   0   0]
 [  0   0  51 371  18   0   0]
 [  0   0   0  95  78   3   0]
 [  0   0   0  13   5  17   0]
 [  0   0   0   0   1   0   0]]
```

The accuracy is 69%; the few extreme observations (4 wines of quality 3 and one wine of quality 9) has been wrongly classified (with 2 or 3 level of difference). All the 7 observations labeled as class 4 are effectively class 4 (100% precision), but the proportion of true instances correctly classified as class 4 (recall) is 21%: most of them are indeed classified as class 5 or 6; precision and recall for classes 5 and 6 oscillates around 70%; class 7 has a precision of 76% but a low recall of 44%: most of these wines are classified as class 6; for both class 6 and 7, the wrongly classified observations are off by no more than a level of quality. Class 8 recall is 49%: the classifier gives half of them worst quality.