inferential_stats_project

February 15, 2015

1 Researching Wine Quality from physiochemical properties.

1.1 Introduction

We look at two datasets related to red and white vinho verde wine samples, from the north of Portugal. The goal is to model wine quality based on physicochemical tests.

Vinho verde is a unique product from the Minho (northwest) region of Portugal. Medium in alcohol, is it particularly appreciated due to its freshness (specially in the summer).

The source states that these datasets can be viewed as *classification* or *regression* tasks. The classes are ordered and not balanced (e.g. there are much more normal wines than excellent or poor ones).

If we can establish this relationship, then of significance would be the sommeliers' reputation who perform wine tasting all the time and providing a certificate of merit or grading to sommeliers such that based on the correlation as stated previously, one could establish the quality of wine and wine quality as set by the taster.

This is really interesting to the massive wine market, wine consumers, wine tasters and to the reputation of esteemed wine-breweries! A predictive model developed on this data is expected to provide guidance to vineyards regarding quality and price expected on their produce without heavy reliance on volatility of wine tasters.

1.2 Research Question and Hypothesis

1.2.1 Research Question

A research question relevant to the topic is: **Is there a correlation between physiochemical makeup of the wine and quality of wine?**

For this, we need to explore the physiochemical attributes given in the dataset and analyis the statistical variables to find out if there is any possibility of a strong correlation. If not, we reject the proposition that there is any correlation. Which brings us close to framing the hypothesis.

1.2.2 Framing the Hypothesis

Null Hypothesis H0 : There is no significant relationship between any of the physiochemical constituents and the quality of Red Wine.

Which means the alternative hypothesis should be:

Alternative Hypothesis HA: There is evidence from Pearson's R values that there is a significant relation between any of the physiochemical constituents of and the quality of Red Wine. We accept the alternative hypothesis **HA** if we get a Pearson's R of atleast 0.60.

1.3 Experimental Design

The Dataset contains the following two samples (with sample size):

1. red wine - 1599;

2. white wine - 4898;

We note that specific attributes have been included per sample as given here: Input variables (based on physicochemical tests):

1. fixed acidity
2. volatile acidity
3. citric acid
4. residual sugar
5. chlorides
6. free sulfur dioxide
7. total sulfur dioxide
8. density
9. pH
10. sulphates
11. alcohol

Output variable (based on sensory data):

12. quality (score between 0 and 10)

We also observe that:

1. Quality is an ordinal variable with possible ranking from 1 (worst) to 10 (best)
2. The attribute data across all samples is a continuous variable including the Quality attribute

Lets first sort the tables in a proper format and analyze the data afterwards.

```
In [108]: #import matplotlib
          %automagic
          %matplotlib inline
          import numpy as np
          import pandas
          from pandas import DataFrame

          import os
          import csv

          def read_write_wine_csv(fpath_name, fpath_new_name):
              """
              Takes a raw unformatted csv file as input.
              Returns a sorted by column, formatted csv file as output
              """
              #open the input file for reading csv
              with open(fpath_name) as f:
                  wine = csv.reader(f, delimiter = ';', quoting=csv.QUOTE_NONNUMERIC)

                  #open output file for writing to csv
                  with open(fpath_new_name, 'wb+') as new_wine_file:
                      #get headers with serial no. as first column
                      headers = ['sno'] + list(wine.next())

                      #write the new file with headers as first row using DictWriter
                      csv_dct_writer = csv.DictWriter(new_wine_file, fieldnames = headers)
                      csv_dct_writer.writerow({col:col for col in headers})
                      #write rest of the rows
                      [csv_dct_writer.writerow(dict(zip(headers,[line_num]+line))) for line_num, line i
```

```
Automagic is ON, % prefix IS NOT needed for line magics.
```

```
In [86]: #Get white wine data sorted
         input_csv = os.getcwd() + r'\wine_data\winequality-white.csv'
         output_csv = os.getcwd() + r'\wine_data\new_white_wine.csv'
         read_write_wine_csv(input_csv, output_csv)

         #Get red wine data sorted
         input_csv = os.getcwd() + r'\wine_data\winequality-red.csv'
         output_csv = os.getcwd() + r'\wine_data\new_red_wine.csv'
         read_write_wine_csv(input_csv, output_csv)
```

```
In [188]: #Get dataset for analysis
          white_wine_df = DataFrame.from_csv(open(os.getcwd()+r'\wine_data\new_white_wine.csv'))
          red_wine_df = DataFrame.from_csv(open(os.getcwd()+r'\wine_data\new_red_wine.csv'))
```

Now that we have the data in our comfortable format, it will be interesting to show that there is a relation between physiochemical attributes of wine and quality of wine based on wine tasting.

We take into account all indicators per sample and perform the following exploratory analysis:

1. Univariate Analysis - Finding descriptive statistics for each attribute with plots across samples of all tables;
2. Removing outliers;
3. Pearson's Correlation;
4. Scatterplot among all attributes;

1.4 Results

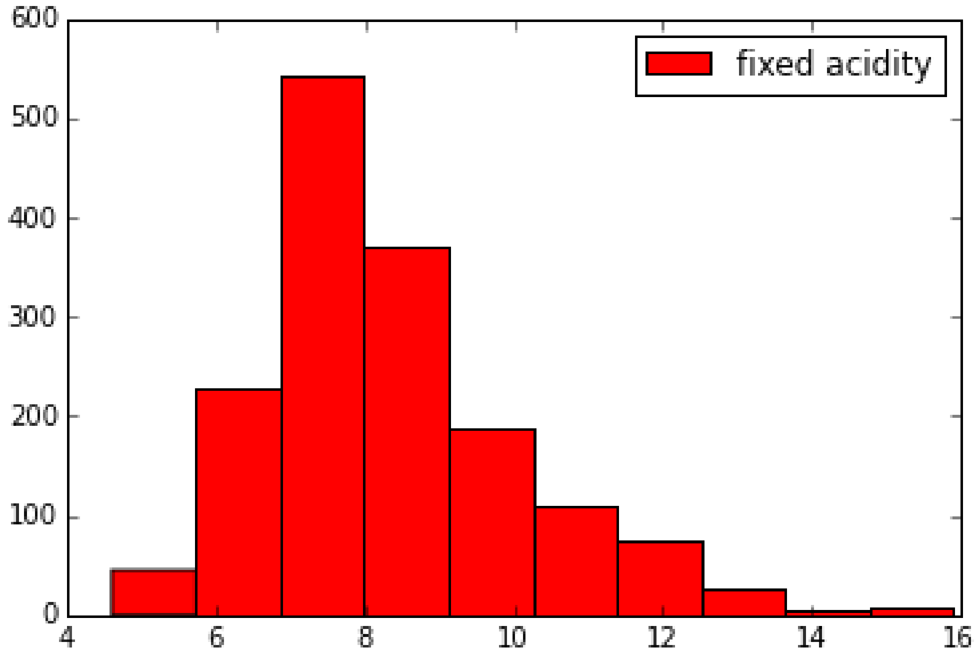
1.4.1 Exploratory Analysis

We take the histogram plots of each attribute in the **Red Wine** dataset.

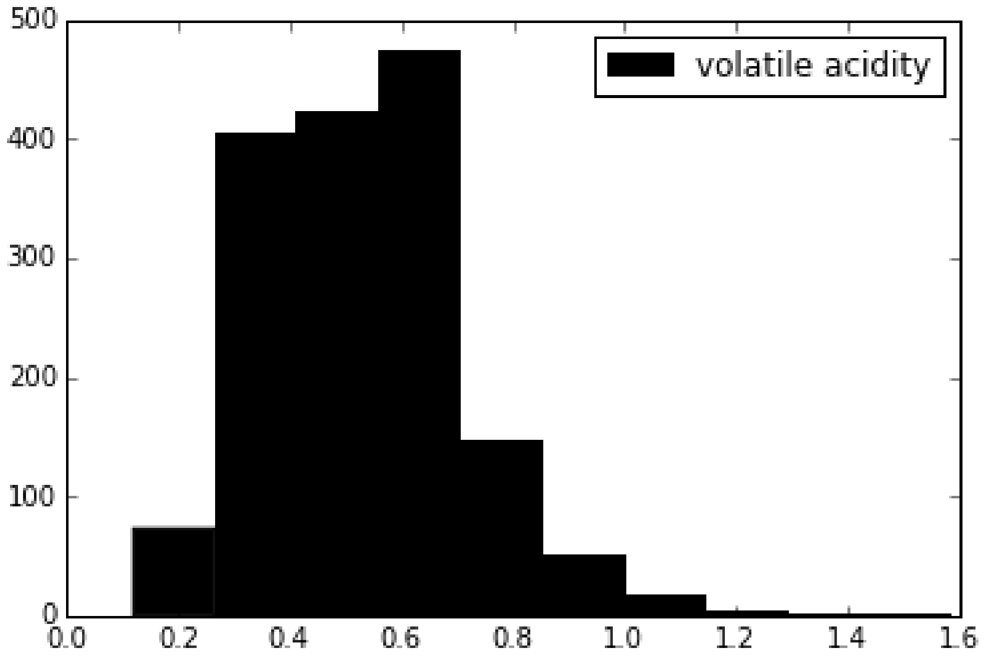
Alongwith taking histogram plots of each attribute, we will clean out any outliers from all the attributes to find out unbiased standard deviation.

```
In [189]: #histogram analysis

          from iqr import quartile_range, outliers_min_max, filter_outlier_df


          #fixed acidity
          fixed_acidity_df = filter_outlier_df(red_wine_df['fixed acidity'])

          hist(red_wine_df['fixed acidity'], color = 'red', label = 'fixed acidity')
          pyplot.legend(loc='upper right')
```

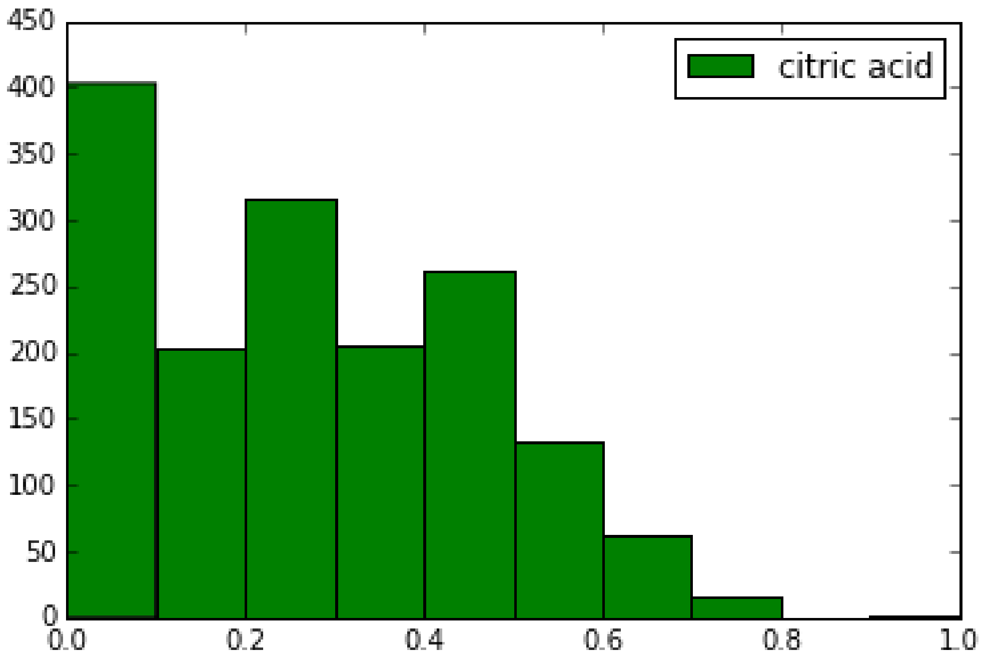
```
Out[189]: <matplotlib.legend.Legend at 0x126a9dd8>
```



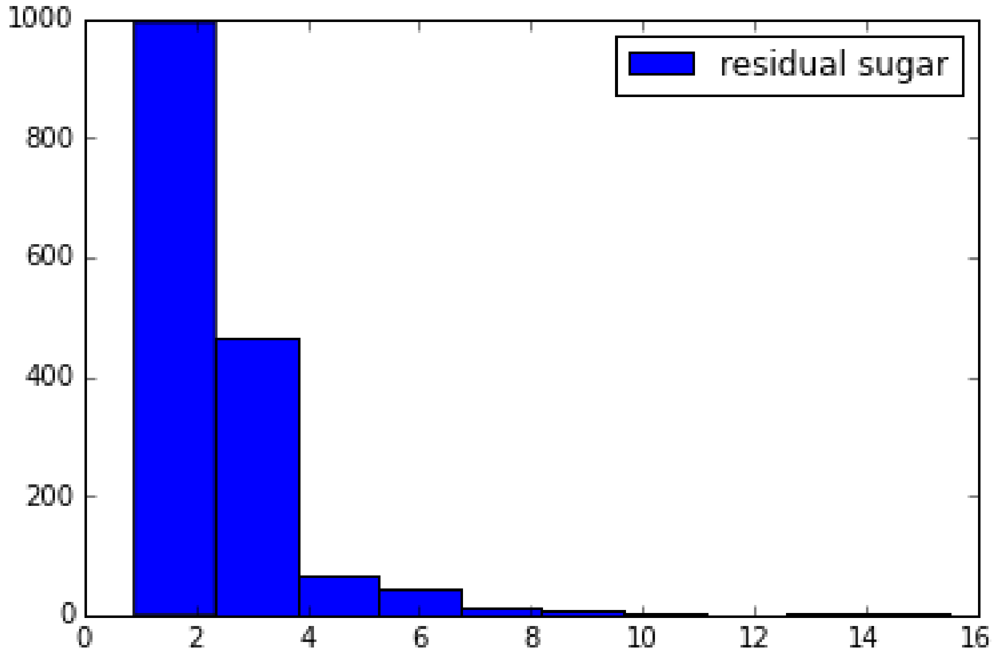
```
In [190]: #volatile acidity
          volatile_acidity_df = filter_outlier_df(red_wine_df['volatile acidity'])

          hist(red_wine_df['volatile acidity'], color = 'black', label = 'volatile acidity')
          pyplot.legend(loc='upper right')

Out[190]: <matplotlib.legend.Legend at 0x130eb358>
```



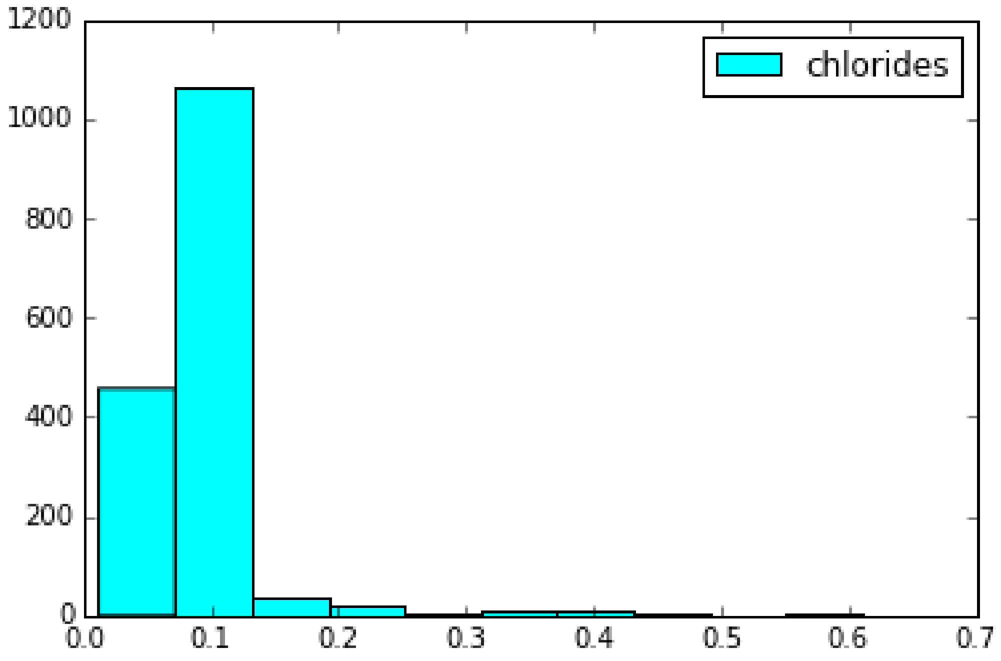
```
In [191]: #citric acid
          citric_acid_df = filter_outlier_df(red_wine_df['citric acid'])

          hist(red_wine_df['citric acid'], color = 'green', label = 'citric acid')
          pyplot.legend(loc = 'upper right')

Out[191]: <matplotlib.legend.Legend at 0x1355b668>
```



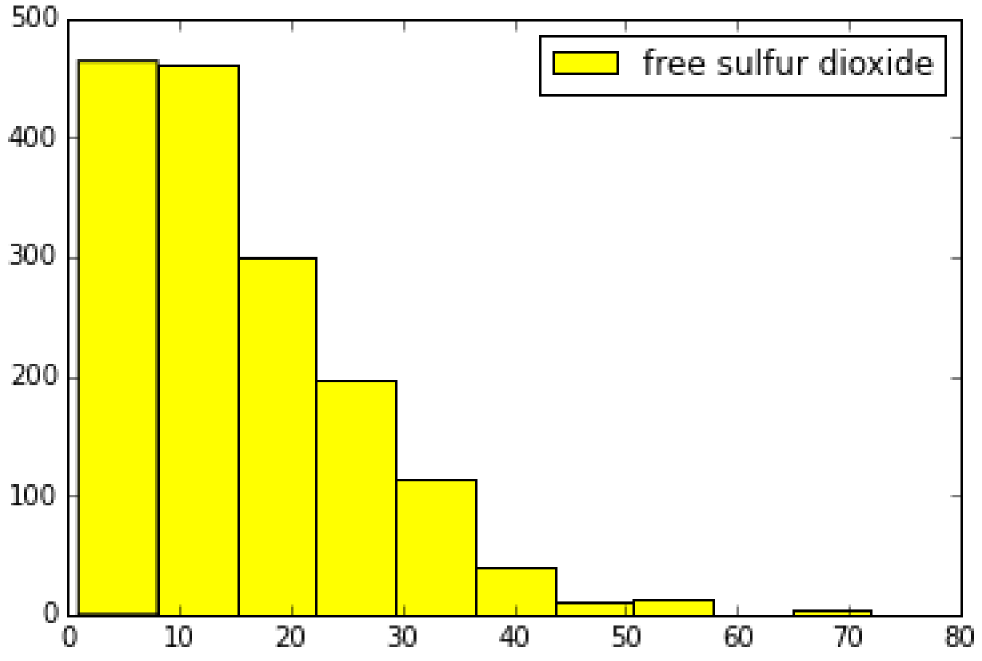
```
In [192]: #residual sugar
          residual_sugar_df = filter_outlier_df(red_wine_df['residual sugar'])

          hist(red_wine_df['residual sugar'], color = 'blue', label = 'residual sugar')
          pyplot.legend(loc = 'upper right')

Out[192]: <matplotlib.legend.Legend at 0x12bc6780>
```



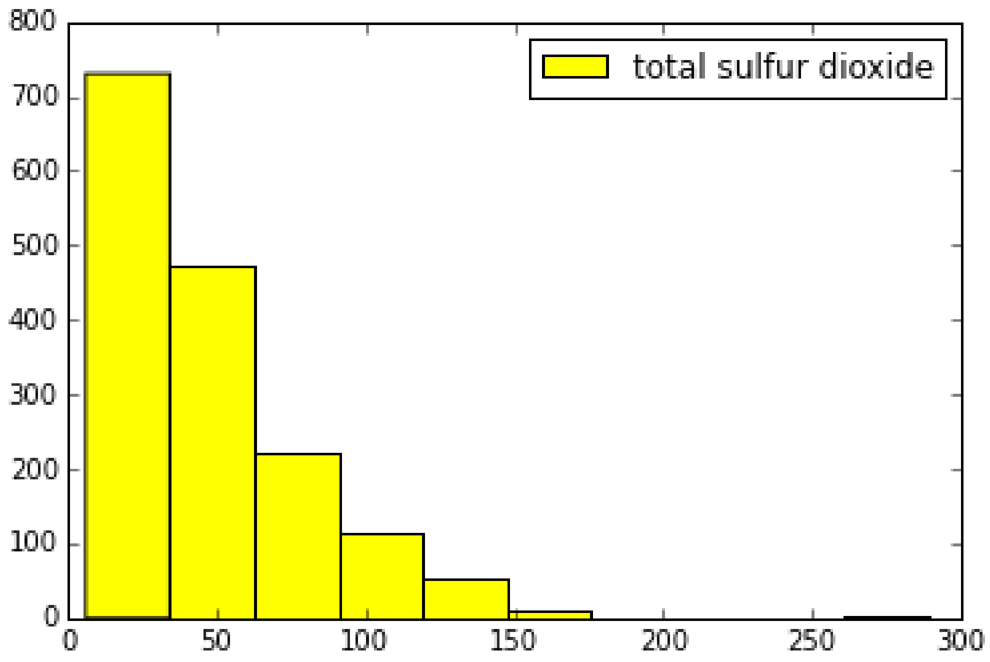
```
In [193]: #chlorides
          chlorides_df = filter_outlier_df(red_wine_df['chlorides'])

          hist(red_wine_df['chlorides'], color = 'cyan', label = 'chlorides')
          pyplot.legend(loc = 'upper right')

Out[193]: <matplotlib.legend.Legend at 0x134380b8>
```



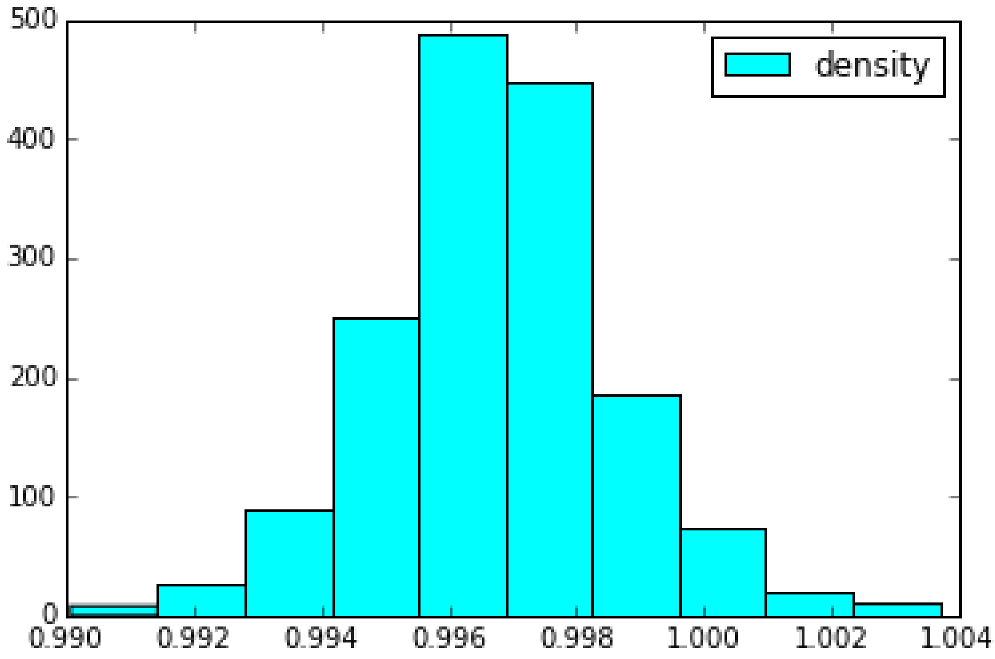
```
In [194]: #free sulfur dioxide
          free_sulfur_dioxide_df = filter_outlier_df(red_wine_df['free sulfur dioxide'])

          hist(red_wine_df['free sulfur dioxide'], color = 'yellow', label = 'free sulfur dioxide')
          pyplot.legend(loc = 'upper right')

Out[194]: <matplotlib.legend.Legend at 0xe032c18>
```



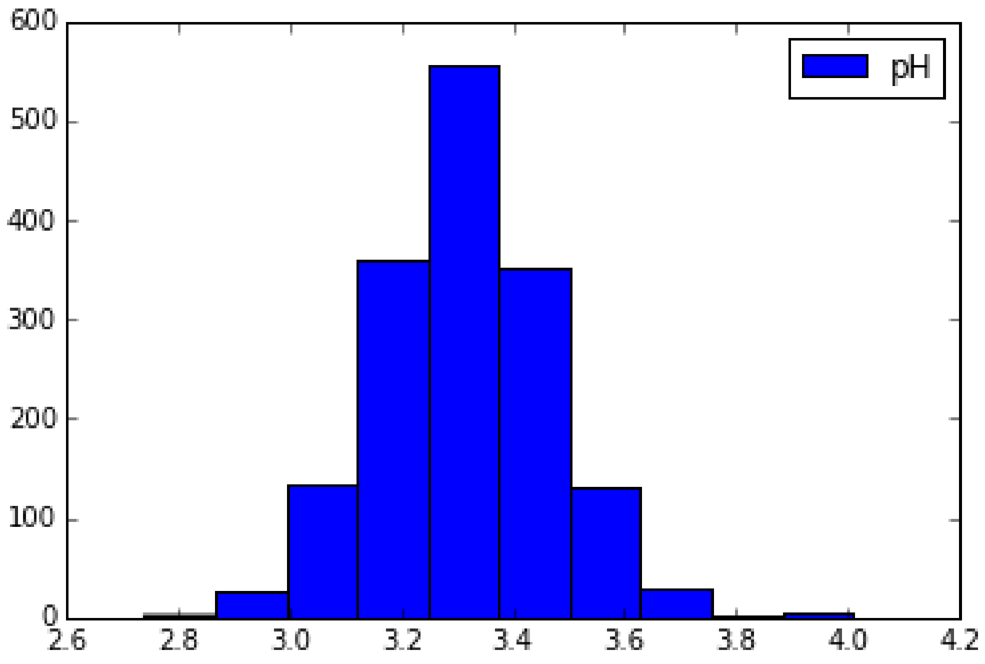
```
In [195]: #total sulfur dioxide
          total_sulfure_dioxide_df = filter_outlier_df(red_wine_df['total sulfur dioxide'])

          hist(red_wine_df['total sulfur dioxide'], color = 'yellow', label = 'total sulfur dioxide')
          pyplot.legend(loc = 'upper right')

Out[195]: <matplotlib.legend.Legend at 0x13438e48>
```



```
In [196]: #density
          density_df = filter_outlier_df(red_wine_df['density'])

          hist(red_wine_df['density'], color = 'cyan', label = 'density')
          pyplot.legend(loc = 'upper right')

Out[196]: <matplotlib.legend.Legend at 0x141e22b0>
```

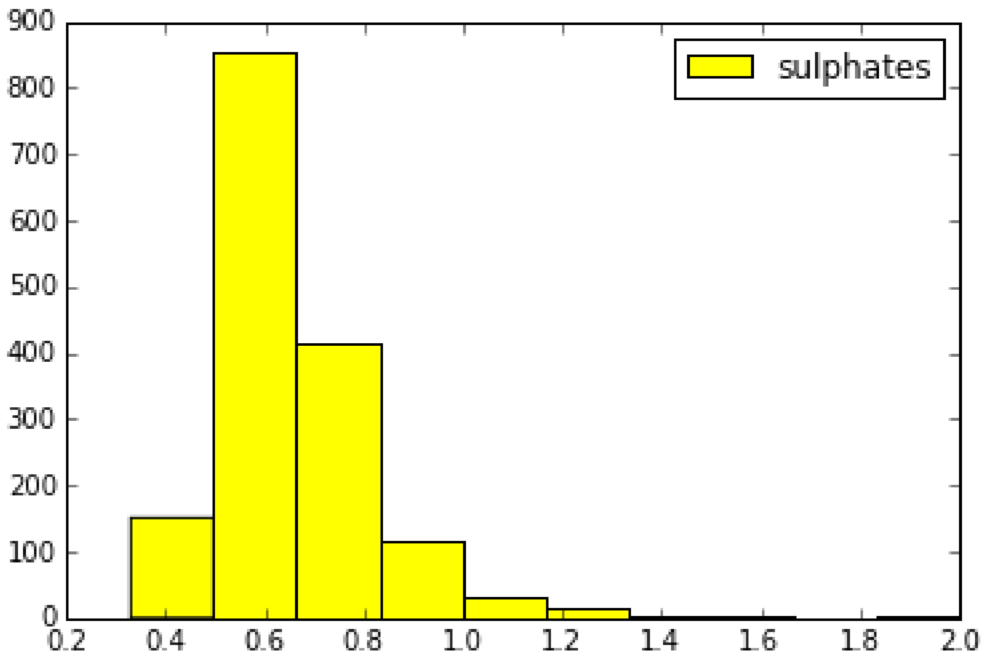


```
In [197]: #pH
          pH_df = filter_outlier_df(red_wine_df['pH'])

          hist(red_wine_df['pH'], color = 'blue', label = 'pH')
          pyplot.legend(loc = 'upper right')

Out[197]: <matplotlib.legend.Legend at 0x13e16f60>
```

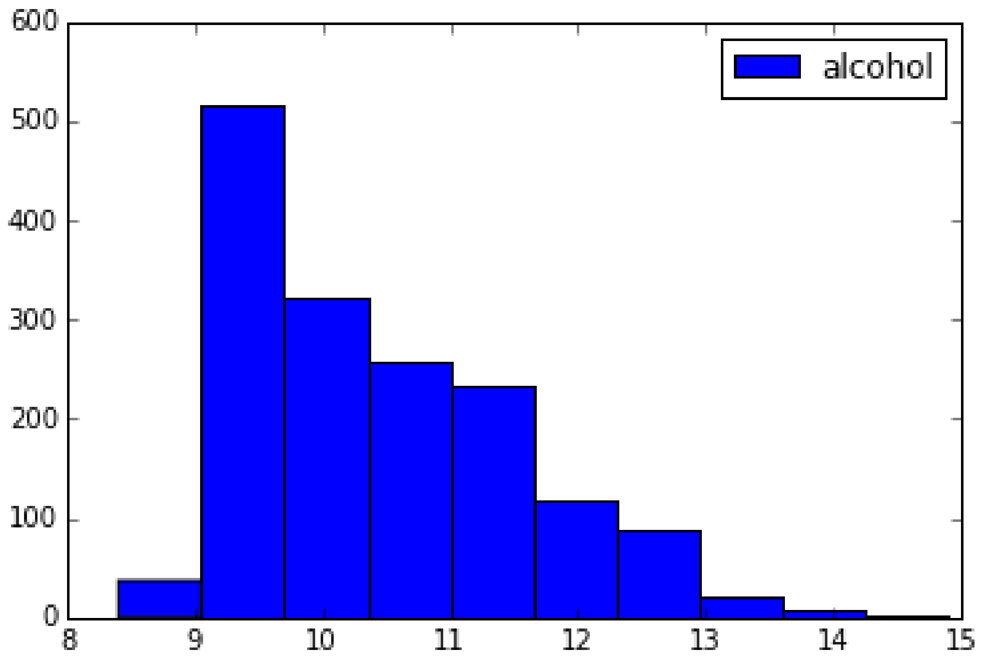


It is interesting to observe the pH values are distributed normally across the data points.

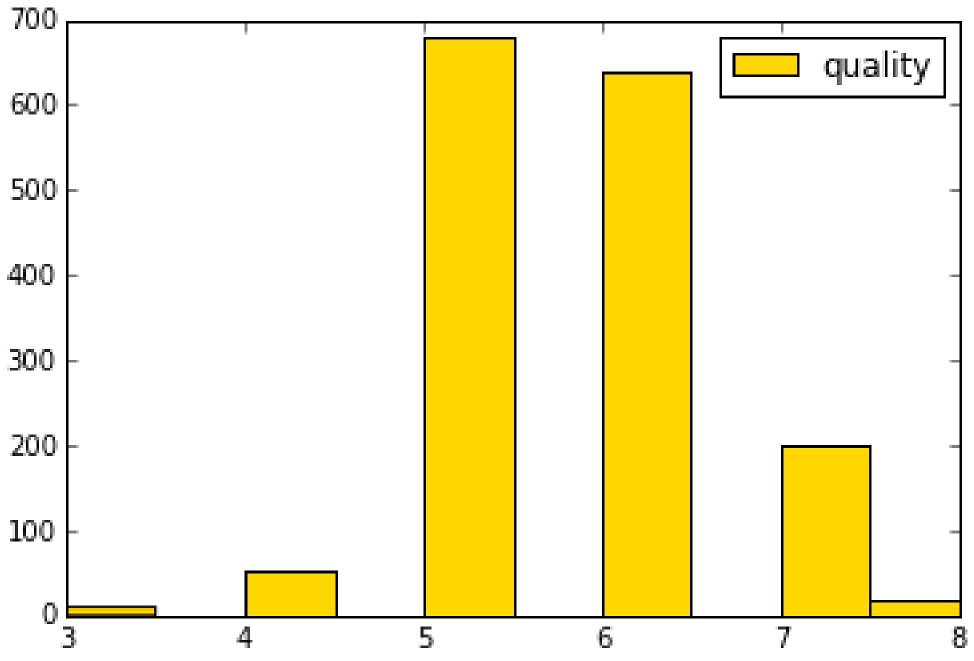
```
In [198]: #sulphates
          sulphates_df = filter_outlier_df(red_wine_df['sulphates'])

          hist(red_wine_df['sulphates'], color = 'yellow', label = 'sulphates')
          pyplot.legend(loc = 'upper right')

Out[198]: <matplotlib.legend.Legend at 0x13999710>
```



```
In [199]: #alcohol
          alcohol_df = filter_outlier_df(red_wine_df['alcohol'])

          hist(red_wine_df['alcohol'], color = 'blue', label = 'alcohol')
          pyplot.legend(loc = 'upper right')

Out[199]: <matplotlib.legend.Legend at 0x13e161d0>
```



```
In [200]: #quality
          quality_df = filter_outlier_df(red_wine_df['quality'])

          hist(red_wine_df['quality'], color = 'gold', label = 'quality')
          pyplot.legend(loc = 'upper right')

Out[200]: <matplotlib.legend.Legend at 0x16626cf8>
```



1.4.2 Statistial & Inferential Analysis

```
In [201]: #find out STD DEV of each of attributes - unbiased Std Dev
          red_wine_std_ds = pandas.Series(red_wine_df.std(), name=['Original Data'])
          red_wine_std_ds.sort()

          #Join the individual filtered attribute series
          quartile_filtered_red_wine_df = pandas.concat([density_df, chlorides_df, pH_df, sulphates_df,
          #find STD DEV of filtered datasets attributes - unbiased Std Dev
          filtered_red_wine_std_ds = pandas.Series(quartile_filtered_red_wine_df.std(), name=['Filtered
          filtered_red_wine_std_ds.sort()

          cmp_std_red_wine_df = DataFrame(data = {'Original STD DEV':red_wine_std_ds, 'Filtered STD DEV

          print "\t\tUnbiased Std Dev\n"
          print cmp_std_red_wine_df

Unbiased Std Dev
```

```
                      Filtered STD DEV  Original STD DEV
density                       0.001667          0.001887
chlorides                     0.014865          0.047065
sulphates                     0.120963          0.169507
pH                            0.140498          0.154386
volatile acidity              0.166581          0.179060
citric acid                   0.194006          0.194801
residual sugar                0.449141          1.409928
quality                       0.745227          0.807569
alcohol                       1.021412          1.065668
fixed acidity                 1.513582          1.741096
free sulfur dioxide           9.226586         10.460157
total sulfur dioxide         27.214797         32.895324
```

We filter the outliers by finding the lower and upper bound of the sorted individual attributes. It can be observed from the **Filtered STD DEV** table that it gives a slimmer standard deviation for :

- density
- chlorides

and increases for other attributes.

This can make for finding correlation between density vs quality and chlorides vs quality or doing a multiregression among density vs chlorides and quality, we still need to investigate further.

Let us take the filtered dataset and perform:

- Univariate analysis;
- Bivariate analysis;

Let us take the quartile filtered data and find the Coefficient of Determination using Pearson's R for all possible combinations for the 11 input attributes as Dataset *df*.

First, let us perform the univariate analysis by finding descriptive statistics variables.

```
In [202]: import math
          from itertools import permutations
          from scipy.stats import pearsonr
          #from linear_regression import r_squared, line_fitting
          #from py_variance_std import critical_t

          red_wine_dof = len(quartile_filtered_red_wine_df) - 2
          #critical_t_val = critical_t(95, red_wine_dof, 0)
          critical_t_per = 0.05

          #Drop all NaN rows
          quartile_filtered_red_wine_df.dropna(inplace=True)
          #Drop all null or nan rows and drop the Quality Column
          quartile_filtered_red_wine_df = quartile_filtered_red_wine_df[quartile_filtered_red_wine_df.n
          [(quartile_filtered_red_wine_df != math.isnan) & (quartile_filtered_red_wine_df != np.NaN)].\
          drop(['quality'], axis=1)

          df = quartile_filtered_red_wine_df

          #get all permutations of column pairs
          keys = df.columns
          attr_permutations = permutations(keys, 2)
```

```
        coeff_r = {}
        [coeff_r.setdefault(each_comb[0], {}).update({each_comb[1]:pearsonr(df[each_comb[0]],\
                                                                          df[each_comb[1]])[0]}) \
         for each_comb in attr_permutations]

        df =  DataFrame(data=coeff_r, columns=keys, index=keys).fillna(1)

        #correlation table
        print df
```

```
density  chlorides        pH  sulphates  \
density               1.000000   0.411902 -0.227471   0.081691
chlorides             0.411902   1.000000 -0.175758  -0.076853
pH                   -0.227471  -0.175758  1.000000   0.013972
sulphates             0.081691  -0.076853  0.013972   1.000000
volatile acidity      0.044174   0.117805  0.220702  -0.317346
citric acid           0.305458   0.073031 -0.470286   0.256747
alcohol              -0.538901  -0.300750  0.126747   0.260783
residual sugar        0.394732   0.232752 -0.054684   0.040172
fixed acidity         0.610125   0.197774 -0.684378   0.165367
free sulfur dioxide  -0.021505   0.012326  0.149126   0.107455
total sulfur dioxide  0.149847   0.176938  0.010031  -0.049655

                      volatile acidity  citric acid   alcohol  residual sugar  \
density                       0.044174     0.305458 -0.538901        0.394732
chlorides                     0.117805     0.073031 -0.300750        0.232752
pH                            0.220702    -0.470286  0.126747       -0.054684
sulphates                    -0.317346     0.256747  0.260783        0.040172
volatile acidity              1.000000    -0.627194 -0.220600        0.035215
citric acid                  -0.627194     1.000000  0.137762        0.149494
alcohol                      -0.220600     0.137762  1.000000        0.098175
residual sugar                0.035215     0.149494  0.098175        1.000000
fixed acidity                -0.271010     0.659397 -0.037999        0.229653
free sulfur dioxide          -0.016718    -0.068852 -0.022415        0.088405
total sulfur dioxide          0.097487     0.004660 -0.245125        0.199534

                      fixed acidity  free sulfur dioxide  total sulfur dioxide
density                    0.610125            -0.021505              0.149847
chlorides                  0.197774             0.012326              0.176938
pH                        -0.684378             0.149126              0.010031
sulphates                  0.165367             0.107455             -0.049655
volatile acidity          -0.271010            -0.016718              0.097487
citric acid                0.659397            -0.068852              0.004660
alcohol                   -0.037999            -0.022415             -0.245125
residual sugar             0.229653             0.088405              0.199534
fixed acidity              1.000000            -0.150845             -0.087083
free sulfur dioxide       -0.150845             1.000000              0.619675
total sulfur dioxide      -0.087083             0.619675              1.000000
```

Now that we have correlation dataset *df*, let us summarize the descriptive analysis in a table.

```
In [203]: from itertools import izip
          from py_variance_std import se

          mean_tbl = quartile_filtered_red_wine_df[['citric acid', 'total sulfur dioxide', 'free sulfur
```

```
        min_tbl = quartile_filtered_red_wine_df[['citric acid', 'total sulfur dioxide', 'free sulfur 
        max_tbl = quartile_filtered_red_wine_df[['citric acid', 'total sulfur dioxide', 'free sulfur 
        #get unbiased std deviation of quartile filtered dataset
        std_tbl = (quartile_filtered_red_wine_df[['citric acid', 'total sulfur dioxide', 'free sulfur
        median_tbl = quartile_filtered_red_wine_df[['citric acid', 'total sulfur dioxide', 'free sulf
        q1_tbl = red_wine_df[['citric acid','total sulfur dioxide','free sulfur dioxide','fixed acidi
        q3_tbl = red_wine_df[['citric acid','total sulfur dioxide','free sulfur dioxide','fixed acidi

        #calculate std err in a dict and create its dataframe
        std_err_dct = {index:se(sd, len(df[index])) \
                              for sd,index in izip(std_tbl, \
                                                   list(std_tbl.index))}
        std_err_tbl = DataFrame(data=std_err_dct, index = ['std err']).transpose()

        #create a DF of all tables and join std_err_tbl
        tbl = DataFrame(data=[mean_tbl, min_tbl, max_tbl, std_tbl, median_tbl, q1_tbl, q3_tbl], index

        #calculate range b/w max and min values
        data_range = DataFrame(data = tbl['max'] - tbl['min'], columns=['range'])
        #calculate Quartile range between Q3 and Q1
        iqr = DataFrame(data = tbl['Q3'] - tbl['Q1'], columns=['iqr'])
        #join tbl with range and Quartile data
        tbl = tbl.join([data_range, iqr])

        print tbl.transpose()
```

```
citric acid  total sulfur dioxide  free sulfur dioxide  fixed acidity
mean        0.246760             42.268024            15.020356       8.162002
min         0.000000              6.000000             1.000000       5.100000
max         0.730000            122.000000            42.000000      12.300000
std         0.179441             26.106438             8.792916       1.458270
median      0.240000             36.000000            13.000000       7.800000
Q1          0.090000             22.000000             7.000000       7.100000
Q3          0.420000             62.000000            21.000000       9.200000
std err     0.054103              7.871387             2.651164       0.439685
range       0.730000            116.000000            41.000000       7.200000
iqr         0.330000             40.000000            14.000000       2.100000
```

We observe that:

- We see that **range** is *greater* than the **IQR** ;
- Overall since unbiased Std Dev of Citric acid is smallest under analysis, it has lowest overall score across all parameters;
- Median is greater than the mean;

We conclude that since range is greater than iqr even after filtering out the outliers, the quartile range is lesser than the range of extreme data points which means that we need to reduce the range less than *iqr*. This implies that we still need to clean the outliers.

Now lets look at the high correlation value r, in the range 0.40 <= r < 1 and see which attributes share these values.

```
In [204]: #High correlations (≥ 40% in absolute value) are identified
        limit_df = df[(df >= .40) & (df < 1)]

        red_wine_correl = {}
```

```
          for each in permutations(keys, 2):
              col, attr = each
              #condition to negate nan/null
              if limit_df[col][attr] > 0.:
                  if attr in red_wine_correl:
                      if (col in red_wine_correl[attr]):
                          continue
                  red_wine_correl.setdefault(col, []).append(attr)

          print red_wine_correl
```

```
{'citric acid': ['fixed acidity'], 'free sulfur dioxide': ['total sulfur dioxide'], 'density': ['chloric
```

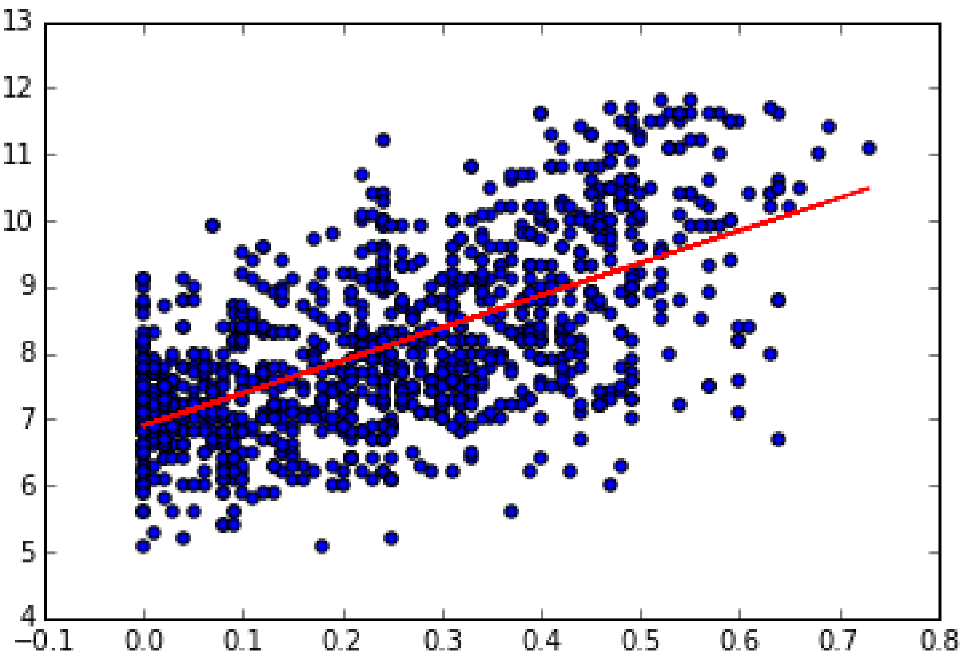
We find out that there seems to be correlation between *Citric acid* and *Fixed acidity*, *density* and *fixed acidity* and *density* and *chlorides* and between free and total *sulfur dioxide*.

Here are the scatter plots for these attributes, using the filtered dataset for red wine.

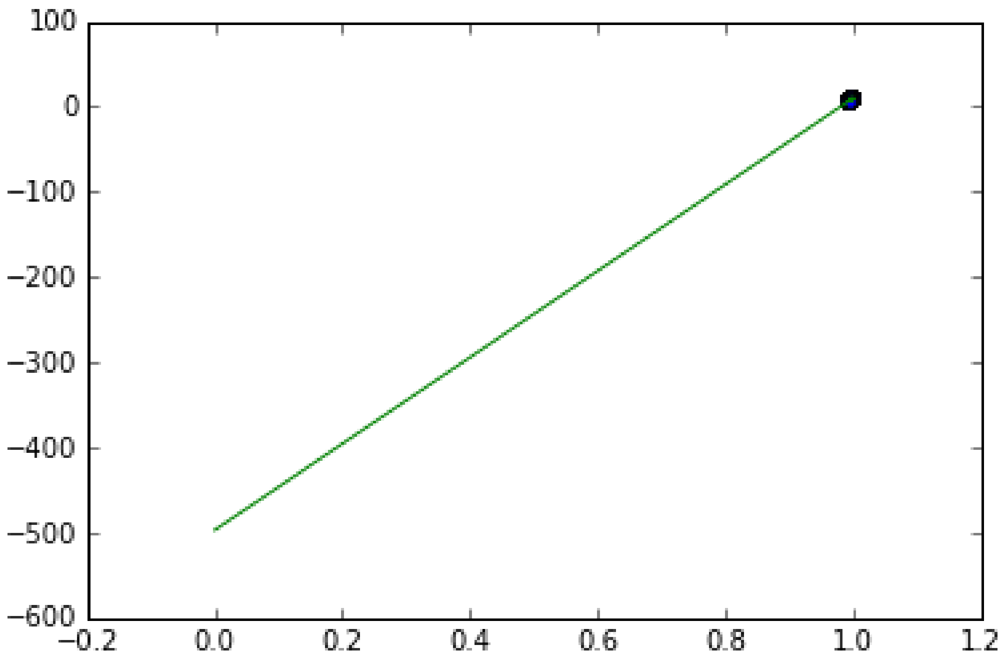
```
In [231]: from linear_regression import trace_line

          #Generate Scatter Plots of the red_wine_correl attributes

          #scatter plot for citric acid and fixed acidity
          #filter out the outliers and recreate DF and drop all null values
          citric_fixed_acid_filter_df = DataFrame(data=[filter_outlier_df(quartile_filtered_red_wine_df
          filter_outlier_df(quartile_filtered_red_wine_df['fixed acidity'])]).transpose().dropna()

          scatter(citric_fixed_acid_filter_df['citric acid'],citric_fixed_acid_filter_df['fixed acidity

          #R value
          citric_fixed_acid_r = pearsonr(citric_fixed_acid_filter_df['citric acid'],citric_fixed_acid_f
          print citric_fixed_acid_r

          #plot line
          x,y = zip(*trace_line(citric_fixed_acid_filter_df['citric acid'],citric_fixed_acid_filter_df[
          plot(x,y, color='red')
```

```
(0.62990096553285213, 1.0407110522614968e-128)
```

```
Out[231]: [<matplotlib.lines.Line2D at 0x18f7a518>]
```



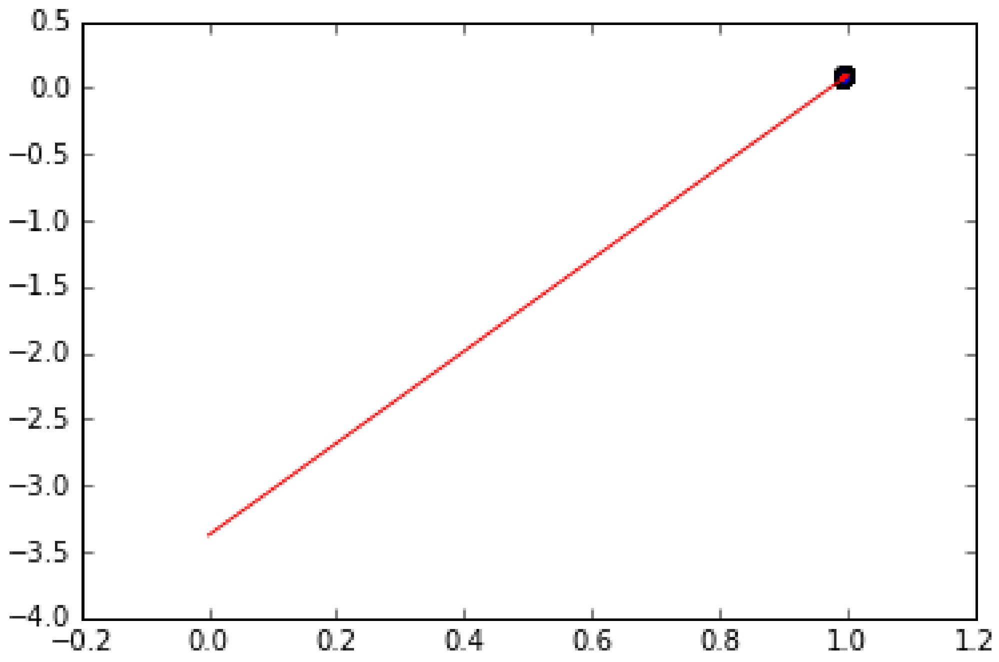
```
In [233]: #filter out the outliers and recreate DF and drop all null values
          density_fixed_acid_filter_df = DataFrame(data=[filter_outlier_df(quartile_filtered_red_wine_d
          filter_outlier_df(quartile_filtered_red_wine_df['fixed acidity'])]).transpose().dropna()

          #scatter plot for density and fixed acidity
          scatter(density_fixed_acid_filter_df['density'],density_fixed_acid_filter_df['fixed acidity']

          #R value
          density_fixed_acid_r = pearsonr(density_fixed_acid_filter_df['density'],density_fixed_acid_fi
          print density_fixed_acid_r

          #plot line
          x,y = zip(*trace_line(density_fixed_acid_filter_df['density'],density_fixed_acid_filter_df['f
          plot(x,y, color='green')

(0.57429600747086051, 7.037170515895179e-102)

Out[233]: [<matplotlib.lines.Line2D at 0x18f7aba8>]
```



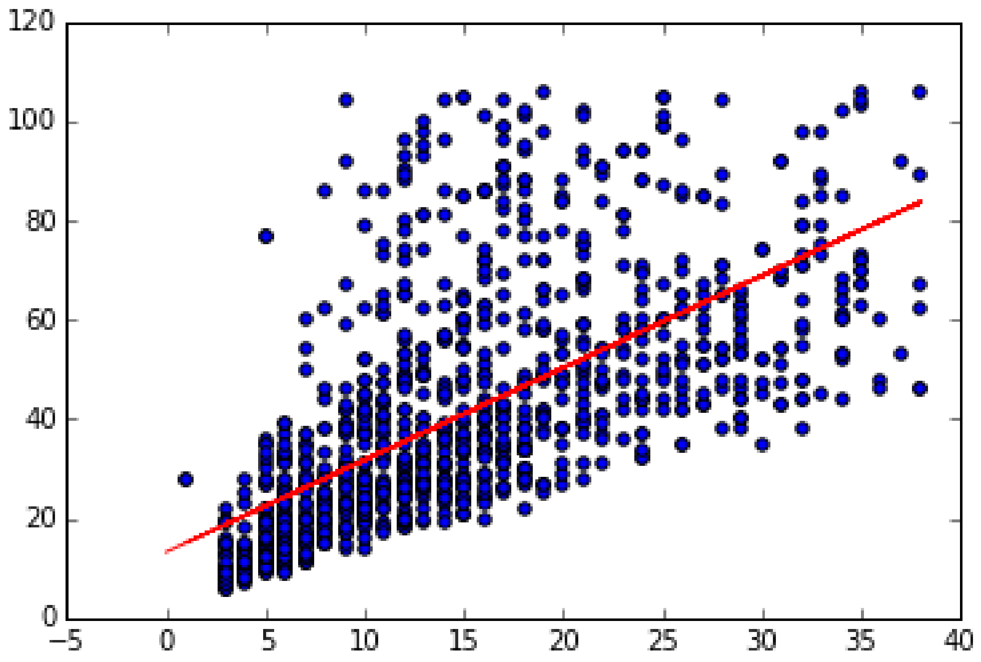
```
In [234]: #filter out the outliers and recreate DF and drop all null values
          density_chlorides_filter_df = DataFrame(data=[filter_outlier_df(quartile_filtered_red_wine_df
          filter_outlier_df(quartile_filtered_red_wine_df['chlorides'])]).transpose().dropna()

          #scatter plot for density and chlorides
          scatter(density_chlorides_filter_df['density'], density_chlorides_filter_df['chlorides'])

          #R value
          density_chlorides_r = pearsonr(density_chlorides_filter_df['density'],\
                                         density_chlorides_filter_df['chlorides'])
          print density_chlorides_r

          #plot line
          x,y = zip(*trace_line(density_chlorides_filter_df['density'], density_chlorides_filter_df['ch
          plot(x,y, color='red')
```

```
(0.41457755335484764, 6.6363192636106639e-49)
```

```
Out[234]: [<matplotlib.lines.Line2D at 0x184f10b8>]
```



```
In [235]: sulfur_r_filter_df = DataFrame(data=[filter_outlier_df(quartile_filtered_red_wine_df['free su
          filter_outlier_df(quartile_filtered_red_wine_df['total sulfur dioxide'])]).transpose().dropna

          #scatter plot for free sulfur dioxide and total sulfur dioxide
          scatter(sulfur_r_filter_df['free sulfur dioxide'], sulfur_r_filter_df['total sulfur dioxide']

          #R value
          sulfur_r = pearsonr(sulfur_r_filter_df['free sulfur dioxide'],\
                              sulfur_r_filter_df['total sulfur dioxide'])
          print sulfur_r

          #plot line
          x,y = zip(*trace_line(sulfur_r_filter_df['free sulfur dioxide'], sulfur_r_filter_df['total su
          plot(x,y, color='red')

(0.65084798161654911, 7.7572802379555849e-138)

Out[235]: [<matplotlib.lines.Line2D at 0x17f84550>]
```



Noticeable correlations:

- ~63% of the relation can be described between citric and fixed acidity;
- ~65% of the relation can be described between free and total sulfur;

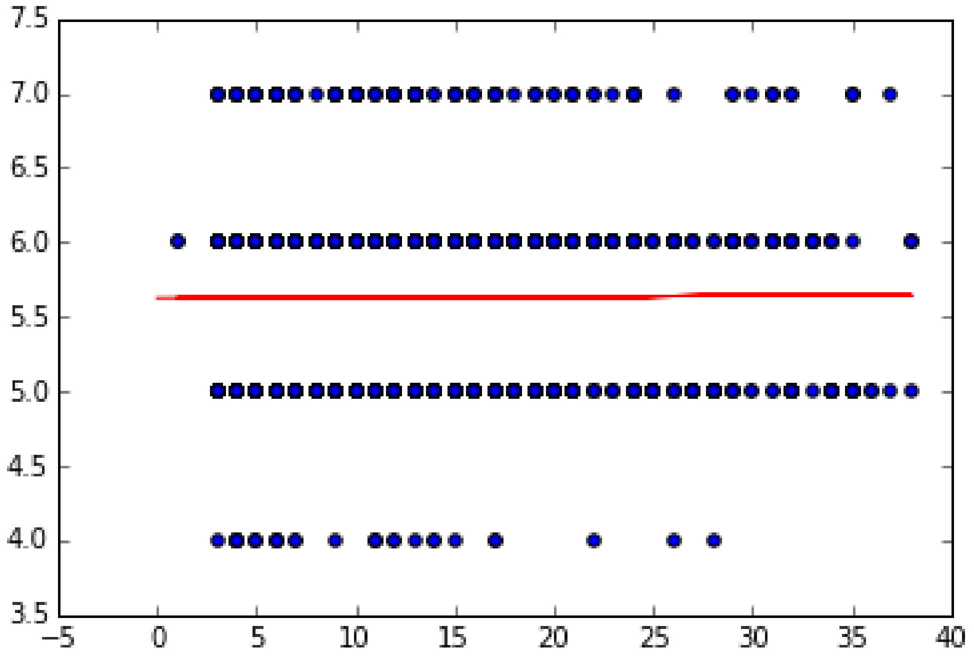
Let us test the scatterplot between:

- Quality vs Free Sulfur Dioxide;
- Quality vs Total Sulfur Dioxide;
- Quality vs Citric acid;
- Quality vs Fixed acidity;

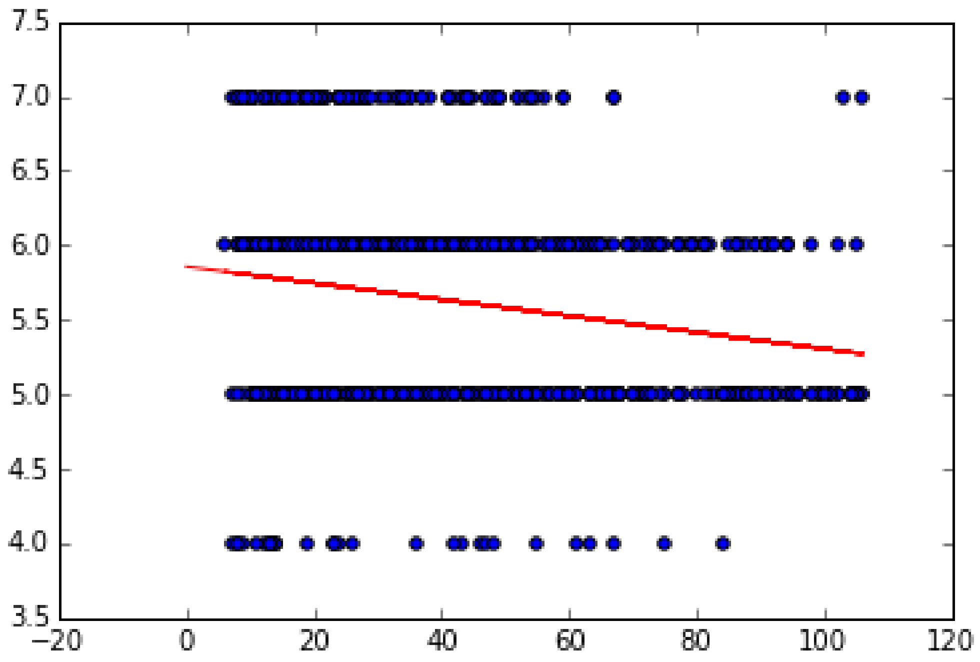
```
In [236]: #generate a DF for all above filtered dataframes with quality attribute
          filtered_df = {attr_name:filter_outlier_df(quartile_filtered_red_wine_df[attr_name]) \
                         for attr_name in df.columns.tolist()}

          filtered_df = DataFrame(data=filtered_df).join(red_wine_df['quality']).dropna()

          #generate scatterplots

          scatter(filtered_df['free sulfur dioxide'], filtered_df['quality'])
          print pearsonr(filtered_df['free sulfur dioxide'], filtered_df['quality'])

          #plot line
          x,y = zip(*trace_line(filtered_df['free sulfur dioxide'], filtered_df['quality']))
          plot(x,y, color='red')
```

```
(0.0057011908643931223, 0.85360463743600268)
```

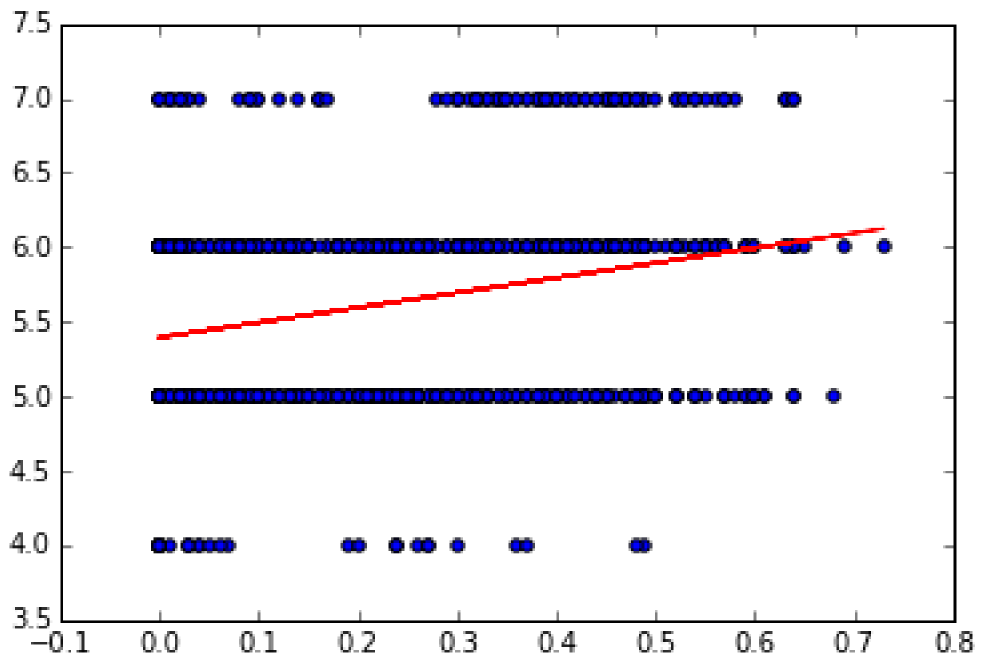
```
Out[236]: [<matplotlib.lines.Line2D at 0x191d92b0>]
```



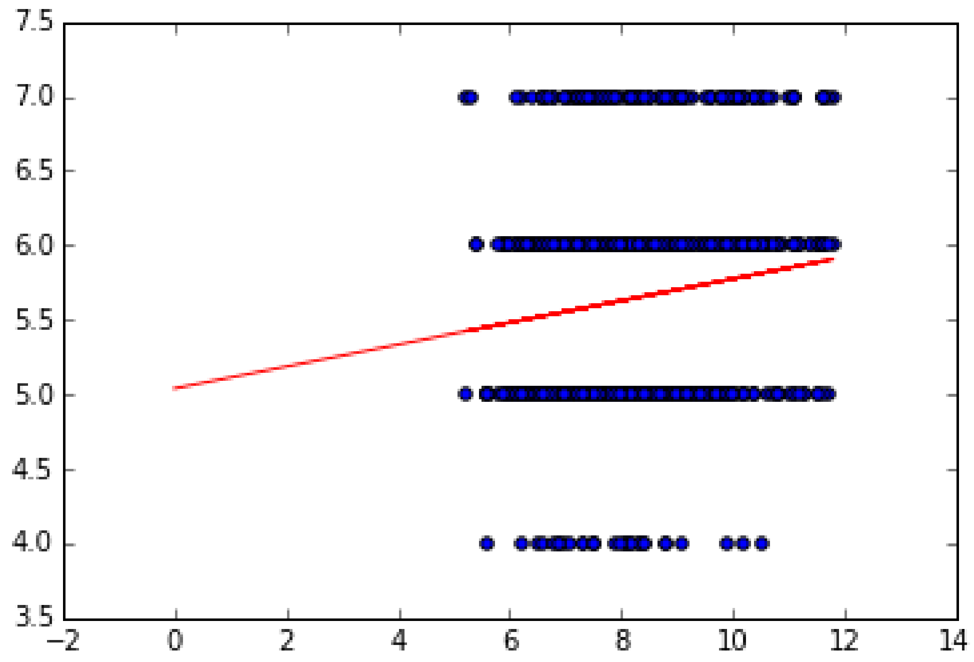
```
In [237]: scatter(filtered_df['total sulfur dioxide'], filtered_df['quality'])
          print pearsonr(filtered_df['total sulfur dioxide'], filtered_df['quality'])

          #plot line
          x,y = zip(*trace_line(filtered_df['total sulfur dioxide'], filtered_df['quality']))
          plot(x,y, color='red')

(-0.18103961670798077, 3.4575535685790347e-09)

Out[237]: [<matplotlib.lines.Line2D at 0x184f4eb8>]
```



```
In [238]: scatter(filtered_df['citric acid'], filtered_df['quality'])
          print pearsonr(filtered_df['citric acid'], filtered_df['quality'])

          #plot line
          x,y = zip(*trace_line(filtered_df['citric acid'], filtered_df['quality']))
          plot(x,y, color='red')

(0.24048937947857821, 2.7961543115120893e-15)

Out[238]: [<matplotlib.lines.Line2D at 0x16a4dd68>]
```



```
In [239]: scatter(filtered_df['fixed acidity'], filtered_df['quality'])
          print pearsonr(filtered_df['fixed acidity'], filtered_df['quality'])

          #plot line
          x,y = zip(*trace_line(filtered_df['fixed acidity'], filtered_df['quality']))
          plot(x,y, color='red')

(0.13620309565661698, 9.4722845730326597e-06)

Out[239]: [<matplotlib.lines.Line2D at 0x16a5b1d0>]
```



From the Pearson R of the scatter plots we conclude there is no significant correlation between (fixed acidity, citric acid, total sulfur dioxide, free sulfur dioxide) and quality as the output variable.

We are unable to establish any significance to the positive correlation between citric vs fixed acidity and quality nor with free vs total sulfur and quality.

1.5 Conclusions

We see that density and chlorides provided minimal standard deviations so we proceed further for a univariate and bivariate analysis to test their correlations with other attributes and quality finally.

We find out the Pearson's R and summarize the descriptive analysis. We fid out the data still has outliers from the extreme range and inter-quartile range. We find out the probable lines of best fit could be among citric acid, (having the minimnum standard error out of all probably correlation attributes) free and total sulfur dioxide, density, chlorides and fixed acidity.

As a result, we re run the filtering process to clear out any outliers from these attributes and run a scatter plot among each of the select physiochemical attributes against quality. But we find that the line of best fit and the scatterplot don't quite do justice and are in agreement with the Pearson's R showing no to littler significance between the correlation of the select physiochemical attributes and the quality of wine.

Therefore, we **fail to reject** the **Null Hypothesis H0** and conclude that the dataset given is not appropriate for this test.

1.5.1 Suggestions based on Result

But for ages Sommeliers have been grading wine based on taste and taste is a side-effect of the physiochemical makeup of the wine. Which leaves us with one possibility and that is of wine-maturity and time.

Thus, a Pre- and Post- Time-period analysis is needed where we record the physiochemical data for a considerable period of time, say over 50 years and record the same attributes at the time of brewing wine and after 50 years and what changes have the physiochemical scores have gone through as well as comparing the *Quality* of wine for atleast these two given periods of time.

This way, we will have a Pre- and Post- dataset with the same number of attributes. With this kind of pair of dataset, we need to test the Null Hypothesis that the Quality of wine has remaind the same and the mean Quality of the two datasets is the same while the Alternative Hypothesis would be that there is significant difference between the two mean Quality attributes. Since, there could be 2 or more datasets over an interval of time, a T-test would be a better measure of significance. If we negate the criteria of independent datasets, a One Way ANOVA test could also be performed over datasets over periods of intervals.

If we find there is a significant difference between the quality per dataset, then we could perform a T-test on the physiochemical makeup between the two datasets. From there, we might be able to infer some significant change in the physiochemical property that is correlated to the change in the quality of the wine over the period of time and come to a conclusion.

```
In []:
```