Assignment 1

BIOM 5405 - Winter 2017

Name:	Maryam Kaka 100929992
Date Submitted:	January 16, 2017

All code used to produced plots and obtain values for this assignment was written in python. Four python libraries (`pandas`, `numpy`, `scipy` and `matlibplot`) were also included to aid with different aspects of data manipulation. All the code used can be found in Appendix A. Code output can be found in Appendix B

1.0 - Classifier Scores

(a) Figure I and Figure II show a scatter plot of Q3 accuracy vs. protein length for both the PCI and PSIPRED classifiers respectively

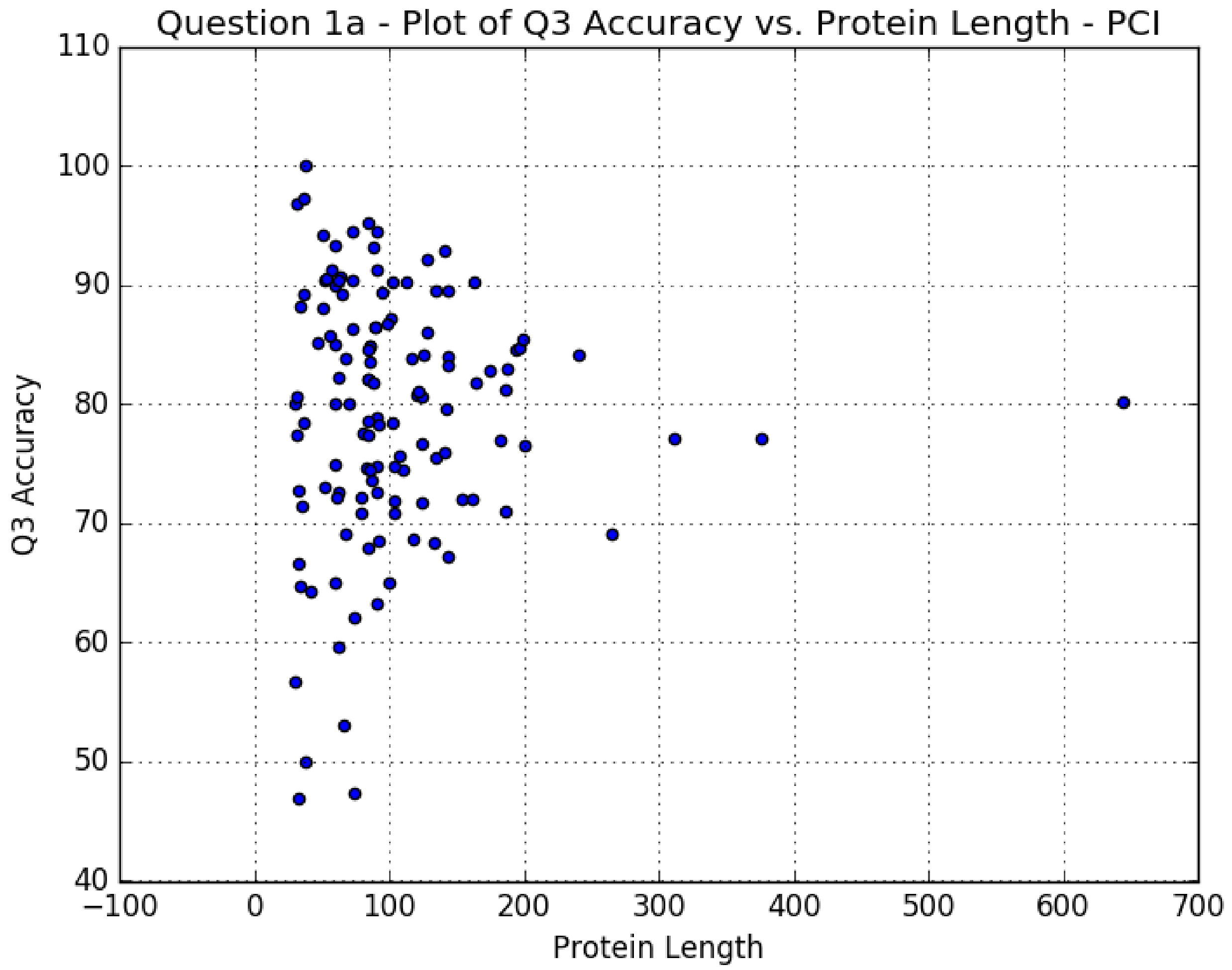

Figure I: Plot of Q3 Accuracy vs. Protein Length for PCI

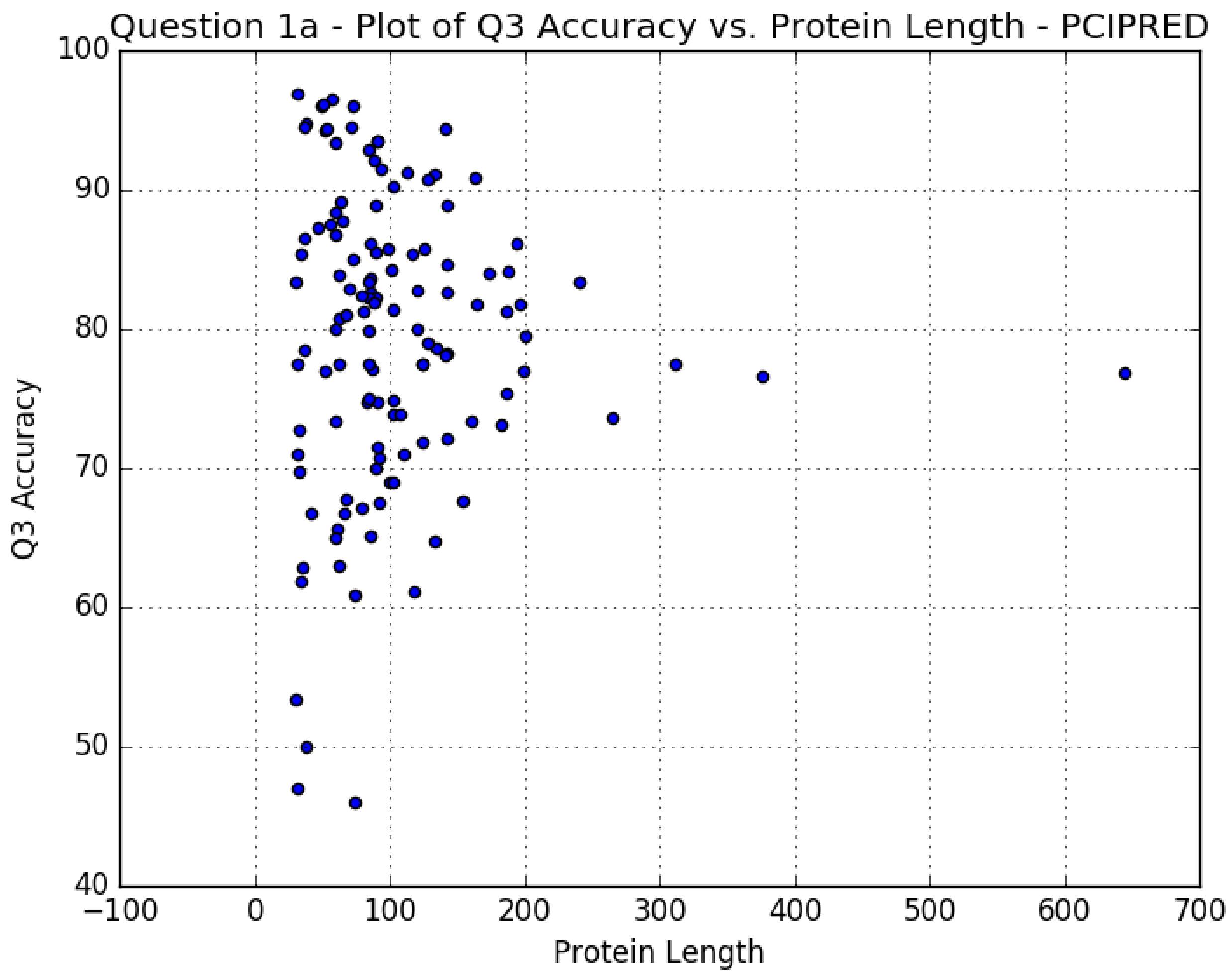

Figure II: Plot of Q3 Accuracy vs. Protein Length for PSIPRED

(b) Table I summarizes the Pearson correlation coefficient calculated between Q3 accuracy and test sequence length. The Pearson correlation coefficient provides an indication of the linear dependence of the two variables (i.e. Q3 accuracy and sequence length). The value of the correlation ranges from -1 to 1 where -1 implies a strong negative linear relationship where as a positive 1 a strong positive. A value close to 0 implies no linear relationship.

Table I: Summary of Pearson Correlation Coefficients between Q3 Accuracy and Sequence Length for each Classifier

Classifier	**Correlation**
PCI	0.02814
PSIPRED	0.010791

(c) Table II shows the mean, median and standard deviation of Q3 accuracy calculated for each method

Table II: Summary statistics of Q3 accuracy for each classification method

Classifier	**Mean**	**Median**	**Standard Deviation**
PCI	79.37	80.65	10.44
PSIPRED	79.35	80.88	10.59

2.0 - Feature Data

(a) Table III summarizes the class-conditional distribution parameters of each feature for each of the classes. The values were calculated using the following equations which describe the parameters for a normal distribution using maximum likelihood estimation [1]:

$$\hat{\mu_n} = \frac{1}{n}\sum_{j=1}^{n} x_j \tag{1}$$

$$\hat{\sigma}_n^2 = \frac{1}{n}\sum_{j=1}^{n}(x_j - \hat{\mu}^2)^2 \tag{2}$$

Where equation 1 is the maximum likelihood estimator for the sample mean and equation 2 is the estimator for the sample variance. It can be noted that using maximum likelihood estimation results in an equation equivalent to the unadjusted parameter [1].

Table III: Summary of class-conditional distribution parameters of each feature

	Weight			**Diameter**		
	Apples	Oranges	Grapes	Apples	Oranges	Grapes
Mean	11.00	11.94	8.73	1006.71	1114.83	832.54
Variance	1.39	6.74	24.52	1605.16	379.57	8272.82

(b) Figures III and IV show the distribution of weights and heights (respectively) measured for each class. The histogram was plotted with 25 bins as it best showed the distributions without distortions. Appendix C shows the effect of varying bin width on both distributions. Looking at the histograms, diameters would be a better feature to select for classification as there is less overlap between each of the histograms. This would allow for boundary selection for each of the classes with minimum error.

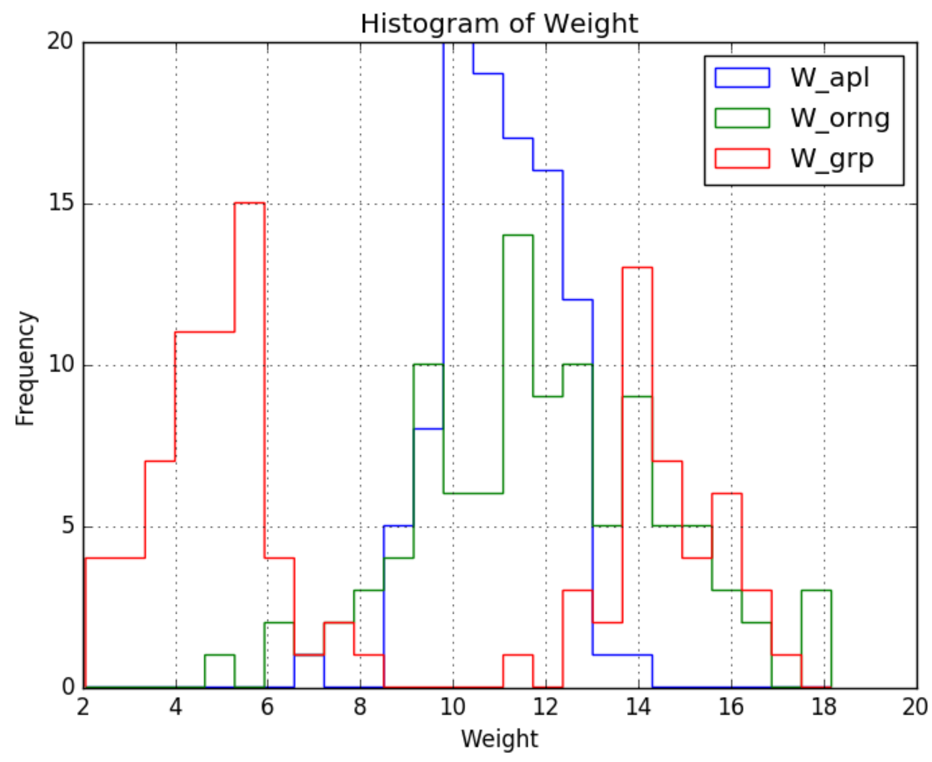

Figure III: Histogram of weights for each fruit plotted with 25 bins

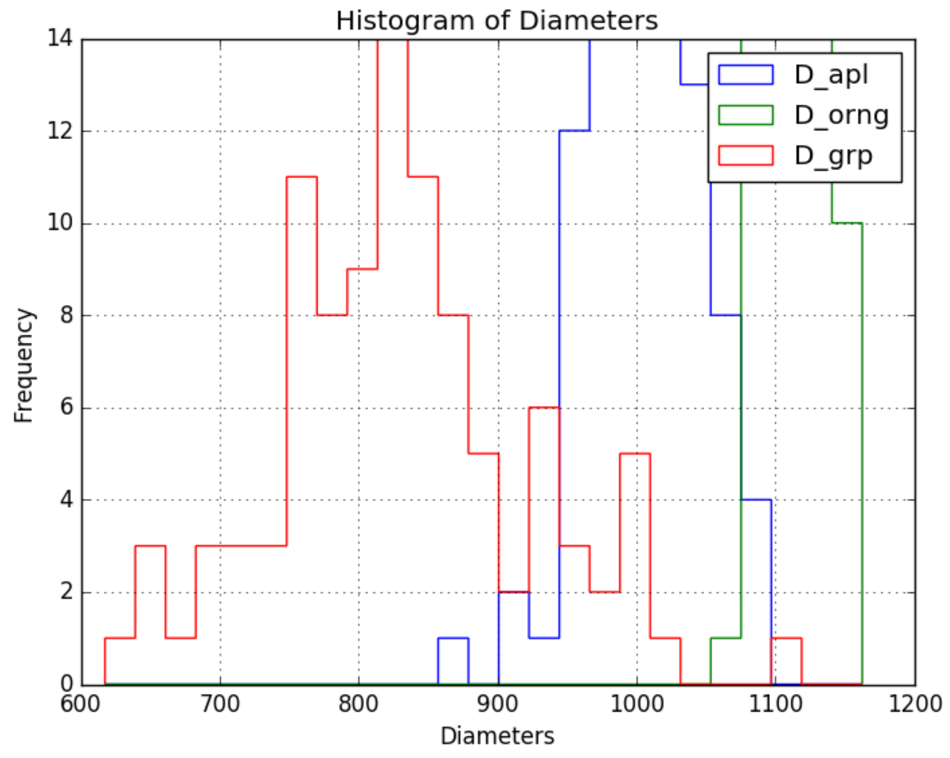

Figure IV: Histogram of diameters for each fruit plotted with 25 bins

(c) Figure V shows the resulting histogram of the combined weight data.

Normality was tested using `normaltest()` found in scipy's statistics library. The `normaltest()` tests whether the sample is normal. The value returned is equivalent to $s^2 + k^2$ where s is a measure of the skew of the curve and k is a measure of the "peakedness" of the distribution as measured by kurtosis test [2]. With both the s and k values a value closer to 0 is equivalent to a normal distribution [3]. Using this test on the combined weight data from all classes resulted in a score of 14.83 which means our distribution is not normal. This is to be expected as if we expect each weight to be somewhat normally distributed then the addition of all three of them would result in a new, non-normal distribution.

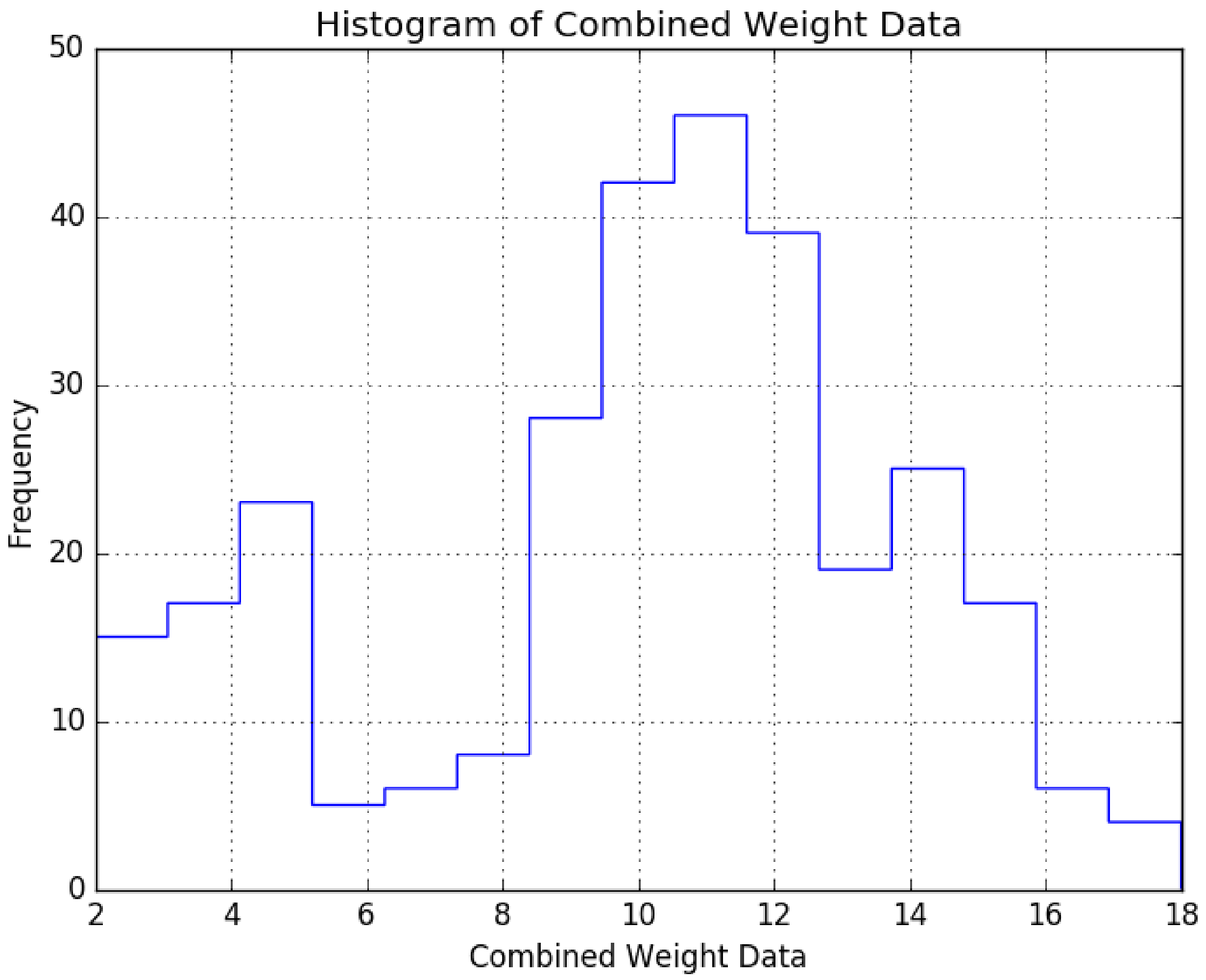

Figure V: Histogram of weights combined for each fruit

3.0 - Generating Data and the Normal Distribution

(b) Figure VI shows a scatter plot of the generated bivariate normal distribution with $\mu_1 = 3.2$, $\mu_2 = 5.1$ and $\Sigma = \left[\begin{smallmatrix} 1.2 & -0.5 \\ -0.5 & 3.3 \end{smallmatrix}\right]$

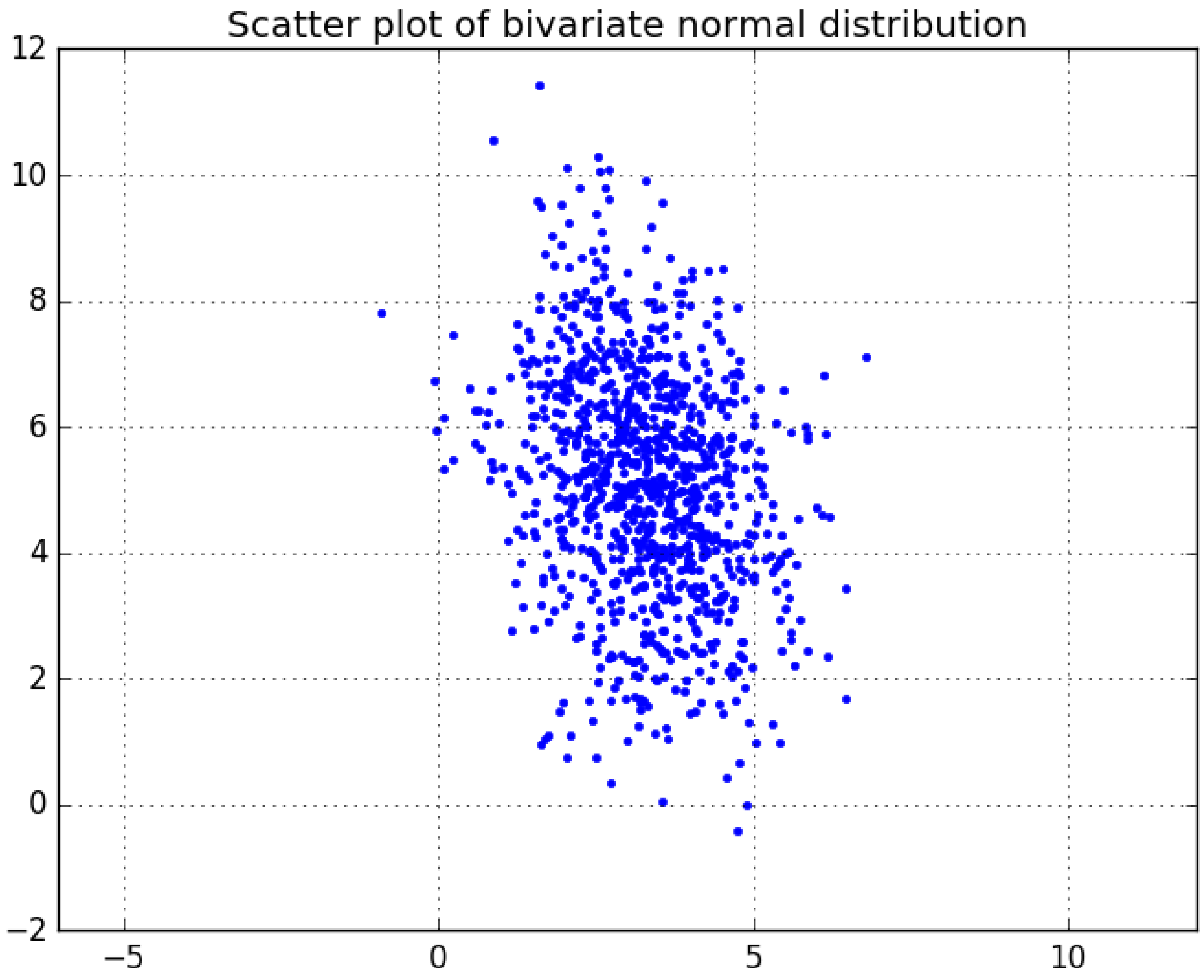

Figure VI: Scatter plot of randomly generated bivariate data

(c) The determinate of the matrix is equal to 3.71 and the trace is 4.5. This matrix is positive definite as $A_1 = 1.2 > 0$ and $det(A_2) = 3.71 > 0$.

(d) The matrix Σ was found to have an the eigenvalues 1.087 and 3.413 which correspond to the eigenvectors $-0.975 + i0.22$ and $-0.22 - i0.975$.

Figure VII shows an ellipse marking a line of equiprobability. The ellipse was generated using code written by Joe Kington [4]. The source code can be found in Appendix A.

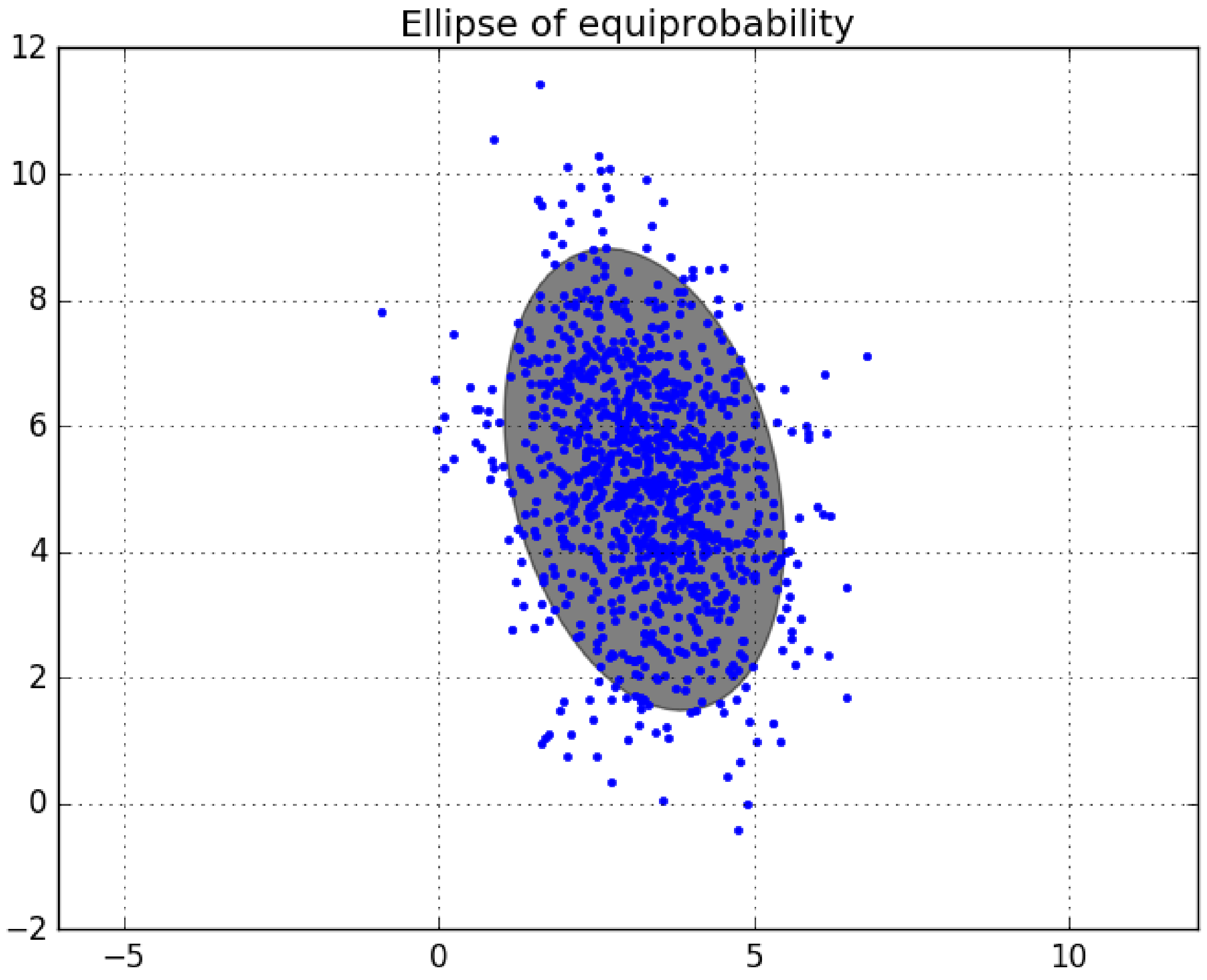

Figure VII: Scatter plot of randomly generated bivariate data showing line of equiprobability

(e) Figure VIII and IX show the probability density function (PDF) and cumulative distribution function (CDF) respectively of a normal distribution with $\mu = 3.2$ and $\hat{\sigma}^2 = 1.2$.

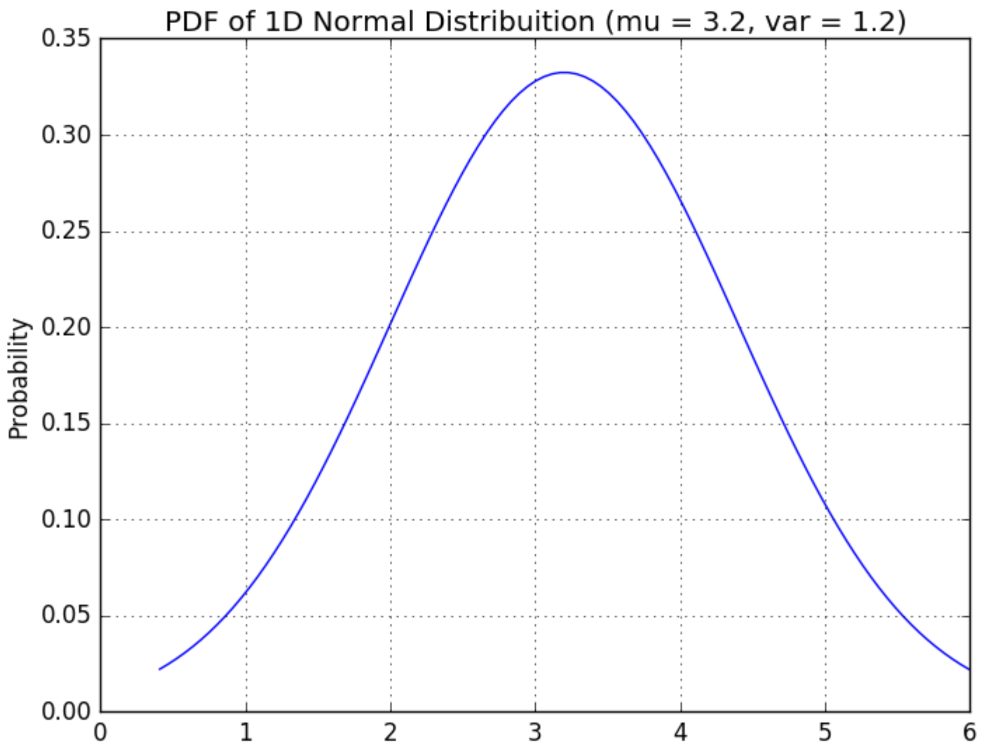

Figure VIII: PDF of normal distribution

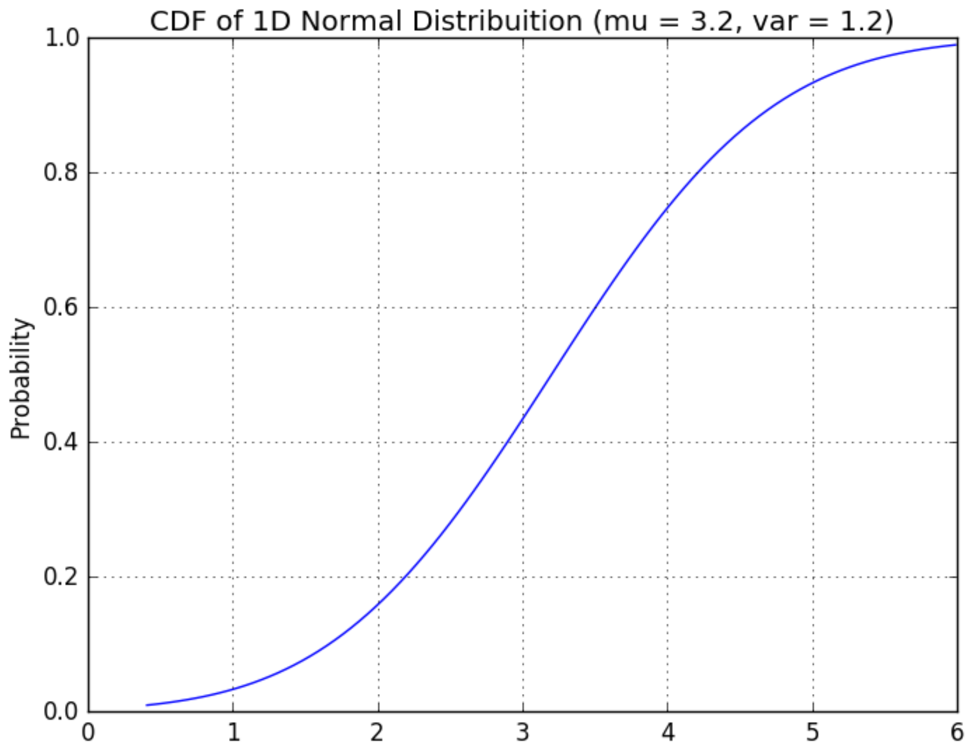

Figure IX: CDF of normal distribution

References

[1] Statlect. Normal distribution - maximum likelihood estimation. [Online]. Available: https://www.statlect.com/fundamentals-of-statistics/normal-distribution-maximum-likelihood

[2] The Scipy community. (2014, May) scipy.stats.mstats.normaltest. [Online]. Available: https://docs.scipy.org/doc/scipy-0.14.0/reference/generated/scipy.stats.mstats.normaltest.html#r246

[3] R. Adams and E. Bogranskaya. (2017) Normality testing - skewness and kurtosis. [Online]. Available: https://help.gooddata.com/display/doc/Normality+Testing+-+Skewness+and+Kurtosis

[4] J. Kington. (2013, December) error_ellipse.py. [Online]. Available: https://github.com/joferkington/oost_paper_code/blob/master/error_ellipse.py

Appendix A - Python Code

```
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np
import scipy.stats as stats
import error_ellipse as ellipse

#variable initialization
outputLocation = 'images/'

def question1(className, data):
    # Part a
    plt.figure()
    plt.scatter(data["Length"], data["Q3"])
    plt.title('Question 1a - Plot of Q3 Accuracy vs. Protein Length - ' +
        className)
    plt.ylabel('Q3 Accuracy')
    plt.xlabel('Protein Length')
    plt.grid(True)
    plt.savefig(outputLocation + 'Question1a-' + className + '.png',
        bbox_inches='tight')

    # Part b
    print("\n(b) Correlation Calculation for " + className)
    print(data[["Q3", "Length"]].corr()) #default Pearson Correlation

    # Part c
    print("\n(c) Statistical Summary for " + className)
    print(data.describe())
```

```
def plotHist(data, xlabel, ylabel, nbin):
    plt.figure()
    data.plot.hist(bins=nbin, histtype='step')
    plt.title('Histogram of ' + xlabel)
    plt.xlabel(xlabel)
    plt.ylabel(ylabel)
    plt.grid(True)
    plt.savefig(outputLocation + xlabel + '-bin' + str(nbin) + '.png',
        bbox_inches='tight')

def main():
    ## Question 1 ##
    print('\nQUESTION 1')
    file = "assigData1.xls"
    question1('PCI', pd.read_excel('assigData1.xls', 0))
    question1('PCIPRED', pd.read_excel('assigData1.xls', 1))

    ## Question 2 ##
    print('\nQUESTION 2')
    file = 'assigData2.tsv'
    data = pd.read_csv(file, sep='\t', index_col=False, header=None,
        names=["W_apl","W_orng", "W_grp", "D_apl", "D_orng", "D_grp"])

    # part a
    print('\n(a) Maximum Likelihood Estimator')
    print('Mean: \n' + str(data.mean()))
    print('Variance: \n' + str(data.var(ddof=0)))   #default is normalized to N-1

    # part b
```

```python
plotHist(data[["W_apl", "W_orng", "W_grp"]], 'Weight', 'Frequency', 10)
plotHist(data[["D_apl", "D_orng", "D_grp"]], 'Diameters', 'Frequency', 10)
plotHist(data[["W_apl", "W_orng", "W_grp"]], 'Weight', 'Frequency', 25)
plotHist(data[["D_apl", "D_orng", "D_grp"]], 'Diameters', 'Frequency', 25)
plotHist(data[["W_apl", "W_orng", "W_grp"]], 'Weight', 'Frequency', 50)
plotHist(data[["D_apl", "D_orng", "D_grp"]], 'Diameters', 'Frequency', 50)

# part c
combined = pd.concat([data["W_apl"], data["W_orng"], data["W_grp"]])
combined = combined.apply(np.floor)
plotHist(combined, 'Combined Weight Data', 'Frequency', 15)
k, p = stats.mstats.normaltest(combined)
print("\n(c) Normal Test: " + str(k) + " (p-value: " + str(p) + ")")

## Question 3 ##
print('\nQUESTION 3')

# part a
mu = [3.2, 5.1]
cov = np.array([[1.2, -0.5], [-0.5, 3.3]])
n = 1000
points = np.random.multivariate_normal(mu, cov, n)
x, y = points.T

# part b
plt.figure()
plt.plot(x, y, 'b.')
plt.axis('equal')
plt.grid(True)
plt.title('Scatter plot of bivariate normal distribution')
plt.savefig(outputLocation+'Question3b.png', bbox_inches='tight')

# part c
print('\n(c)')
print('Det: ' + str(np.linalg.det(cov)))
print('Trace: ' + str(np.matrix.trace(cov)))

# part d
print('\n(d)')
eigenvals, eigenvect = np.linalg.eig(cov)
print('eigenvalues: ' + str(eigenvals))
print('eigenvectors: \n' + str(eigenvect))
ellipse.plot_point_cov(points, nstd=2 , color='k', alpha=0.5)
plt.axis('equal')
plt.grid(True)
plt.title('Ellipse of equiprobability')
plt.savefig(outputLocation + 'ellipse.png', bbox_inches='tight')

# part e
mu = 3.2
var = 1.2
std = math.sqrt(var)
dist = stats.norm(loc=mu, scale=var)
x = np.linspace(dist.ppf(0.01), dist.ppf(0.99), 100)
plt.figure()
plt.title('PDF of 1D Normal Distribuition (mu = ' + str(mu) + ', var = ' +
    str(var) + ')')
plt.ylabel('Probability')
plt.grid(True)
plt.plot(x, dist.pdf(x))
```

```
    plt.savefig(outputLocation + 'pdf.png', bbox_inches='tight')
    plt.figure()
    plt.title('CDF of 1D Normal Distribuition (mu = ' + str(mu) + ', var = ' +
        str(var) + ')')
    plt.ylabel('Probability')
    plt.grid(True)
    plt.plot(x, dist.cdf(x))
    plt.savefig(outputLocation + 'cdf.png', bbox_inches='tight')
main();
```

error_ellipse.py [4]

```
import numpy as np

import matplotlib.pyplot as plt

from matplotlib.patches import Ellipse


def plot_point_cov(points, nstd=2, ax=None, **kwargs):
    """
    Plots an `nstd` sigma ellipse based on the mean and covariance of a point
    "cloud" (points, an Nx2 array).

    Parameters
    ----------
        points : An Nx2 array of the data points.
        nstd : The radius of the ellipse in numbers of standard deviations.
            Defaults to 2 standard deviations.
        ax : The axis that the ellipse will be plotted on. Defaults to the
            current axis.
        Additional keyword arguments are pass on to the ellipse patch.

    Returns
    -------
        A matplotlib ellipse artist
    """
    pos = points.mean(axis=0)
    cov = np.cov(points, rowvar=False)
    return plot_cov_ellipse(cov, pos, nstd, ax, **kwargs)

def plot_cov_ellipse(cov, pos, nstd=2, ax=None, **kwargs):
    """
    Plots an `nstd` sigma error ellipse based on the specified covariance
```

```
matrix (`cov`). Additional keyword arguments are passed on to the
ellipse patch artist.

Parameters
----------
    cov : The 2x2 covariance matrix to base the ellipse on
    pos : The location of the center of the ellipse. Expects a 2-element
        sequence of [x0, y0].
    nstd : The radius of the ellipse in numbers of standard deviations.
        Defaults to 2 standard deviations.
    ax : The axis that the ellipse will be plotted on. Defaults to the
        current axis.
    Additional keyword arguments are pass on to the ellipse patch.

Returns
-------
    A matplotlib ellipse artist
"""
def eigsorted(cov):
    vals, vecs = np.linalg.eigh(cov)
    order = vals.argsort()[::-1]
    return vals[order], vecs[:,order]

if ax is None:
    ax = plt.gca()

vals, vecs = eigsorted(cov)
theta = np.degrees(np.arctan2(*vecs[:,0][::-1]))

# Width and height are "full" widths, not radius
```

```
    width, height = 2 * nstd * np.sqrt(vals)
    ellip = Ellipse(xy=pos, width=width, height=height, angle=theta, **kwargs)

    ax.add_artist(ellip)
    return ellip
```

Appendix B - Code Output

```
QUESTION 1

(b) Correlation Calculation for PCI
             Q3   Length
Q3      1.00000  0.02814
Length  0.02814  1.00000

(c) Statistical Summary for PCI
           Length          CC          Q3         BAD
count  125.000000  125.000000  125.000000  125.000000
mean   103.240000    0.656040   79.370320    1.676856

std     74.924952    0.177874   10.440728    3.206898
min     30.000000    0.137000   46.880000    0.000000
25%     61.000000    0.543000   72.580000    0.000000
50%     87.000000    0.669000   80.650000    0.000000
75%    124.000000    0.786000   86.520000    1.961000
max    644.000000    0.962000  100.000000   18.421000

(b) Correlation Calculation for PCIPRED
              Q3    Length
Q3      1.000000  0.010791
Length  0.010791  1.000000

(c) Statistical Summary for PCIPRED
           Length      CC_AVG          Q3         BAD
count  125.000000  125.000000  125.000000  125.000000
mean   103.240000    0.657968   79.356880    2.198808
```

```
std     74.924952    0.175720   10.595322    3.755219
min     30.000000    0.145000   45.950000    0.000000
25%     61.000000    0.561000   73.290000    0.000000
50%     87.000000    0.673000   80.880000    0.000000
75%    124.000000    0.784000   86.080000    3.226000
max    644.000000    0.944000   96.770000   21.053000

QUESTION 2

(a) Maximum Likelihood Estimator

Mean:
W_apl       11.003084
W_orng      11.944999
W_grp        8.733358
D_apl     1006.707200
D_orng    1114.833850
D_grp      832.546227
dtype: float64

Variance:
W_apl        1.387092
W_orng       6.738552
W_grp       24.539227
D_apl     1605.160717
D_orng     379.571763
D_grp     8272.817843
dtype: float64

(c) Normal Test: 14.8399244321 (p-value: 0.000599171784176)
```

```
QUESTION 3

(c)

Det: 3.71

Trace: 4.5


(d)

eigenvalues: [ 1.08702967  3.41297033]

eigenvectors:

[[-0.97541287  0.22038544]

 [-0.22038544 -0.97541287]]
```

Appendix C - Varying Bin Width

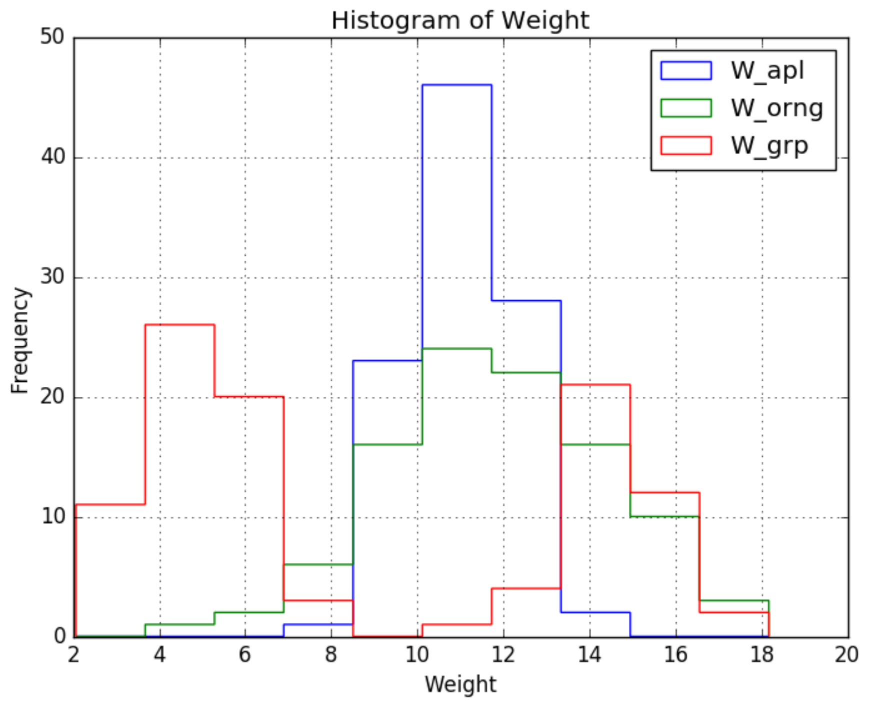

Figure X: Histogram of weights for each fruit plotted with 10 bins

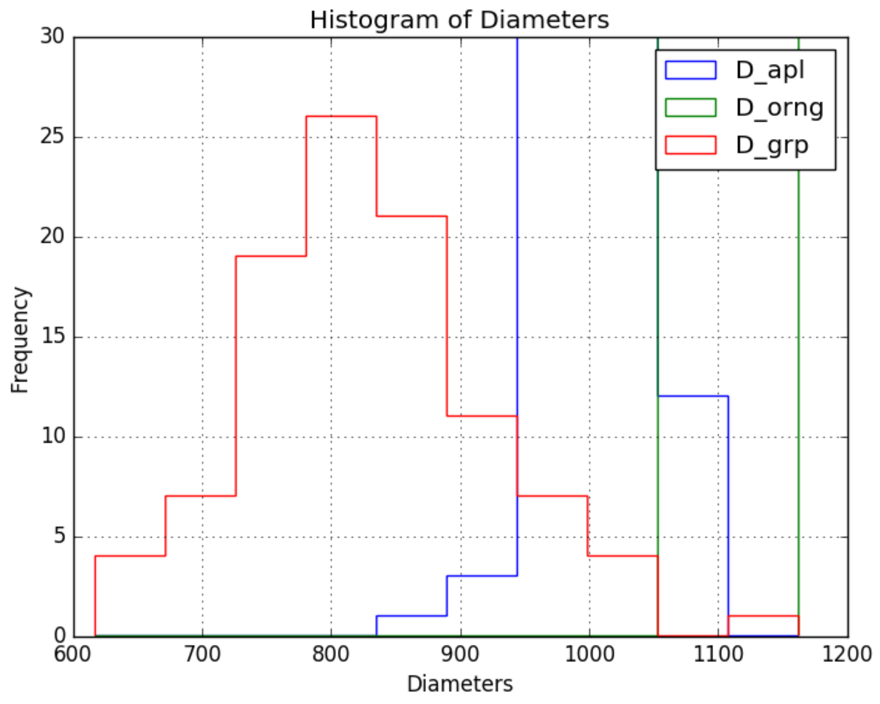

Figure XI: Histogram of diameters for each fruit plotted with 10 bins

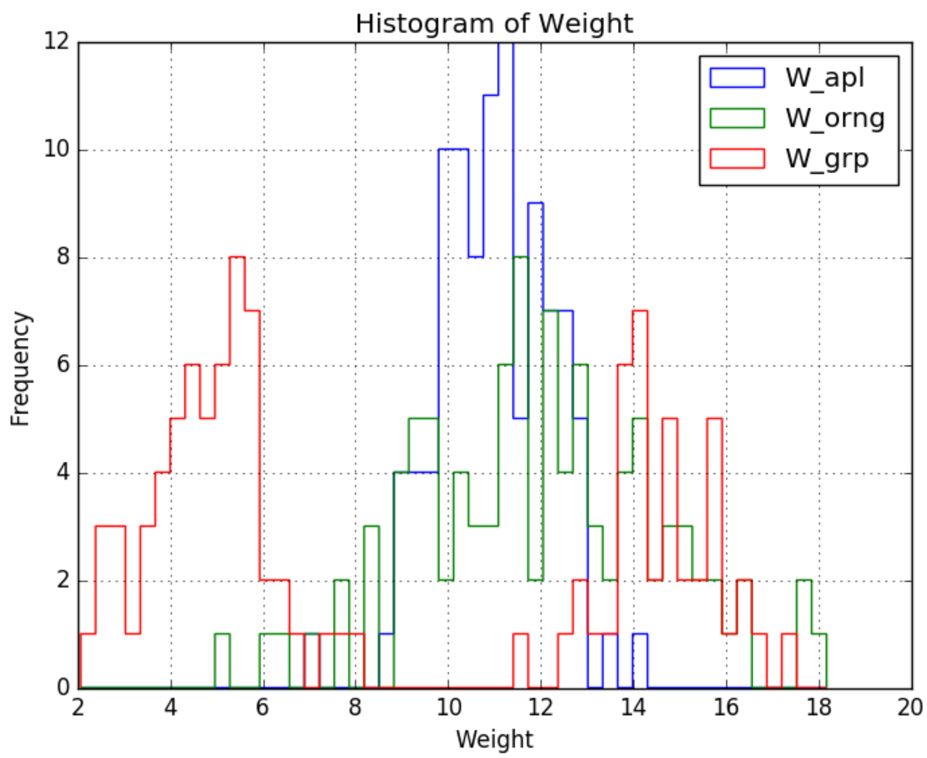

Figure XII: Histogram of weights for each fruit plotted with 50 bins

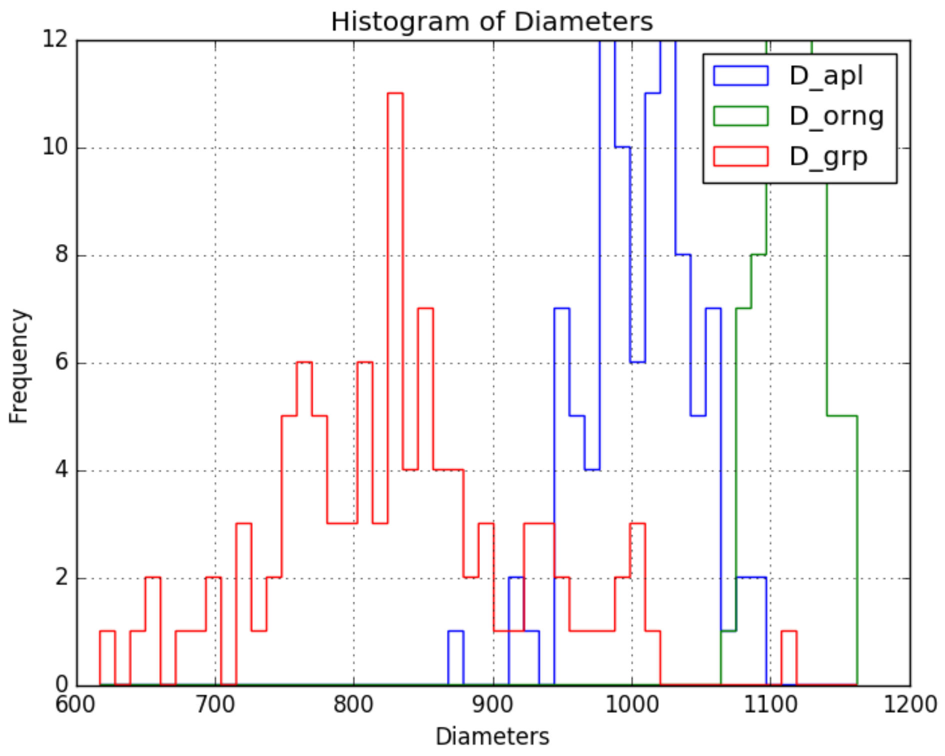

Figure XIII: Histogram of diameters for each fruit plotted with 50 bins