Open in Colab
(https://colab.research.google.com/gist/jeannereppert/9ffa88cbc81a38b907f2cca29416badc/reppert_project1_iaf_604.ipynb)

Step 1 - Install packages and set up spark

In [1]:

```
#make connection to google colab drive
from google.colab import drive
drive.mount('/content/drive')
```

```
Drive already mounted at /content/drive; to attempt to forcibly remount, call drive.mount("/content/d
```

In [0]:

```
!apt-get install openjdk-8-jdk-headless -qq > /dev/null
!wget -q https://archive.apache.org/dist/spark/spark-2.4.5/spark-2.4.5-bin-hadoop2.7.tgz
!tar xf spark-2.4.5-bin-hadoop2.7.tgz
!pip install -q findspark
```

In [0]:

```
import os
os.environ["JAVA_HOME"] = "/usr/lib/jvm/java-8-openjdk-amd64"
os.environ["SPARK_HOME"] = "/content/spark-2.4.5-bin-hadoop2.7"
```

In [0]:

```
import findspark
findspark.init()
from pyspark.sql import SparkSession
spark = SparkSession.builder.master("local[*]").getOrCreate()
```

In [0]:

```
import pyspark
from pyspark import SparkContext
from pyspark.ml import Pipeline
from pyspark.ml.feature import IndexToString, StringIndexer, VectorIndexer
from pyspark.ml.evaluation import MulticlassClassificationEvaluator
from pyspark.sql import *
from pyspark.sql.functions import *
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
```

Step 2 - Read in carwood.csv as cw, file type 'pyspark.sql.dataframe.Dataframe'

In [0]:

```
cw = spark.read.csv('/content/drive/My Drive/Colab IAF 604/carwood.csv',inferSchema=True, header =True)
```

In [7]:

```
#verifying working with a pyspark dataframe
print(type(cw))
```

```
<class 'pyspark.sql.dataframe.DataFrame'>
```

Step 3 - Explore carwood dataframe

In [8]:

```
cw['f1', 'f2'].show(2)
```

```
+------+------+
|    f1|    f2|
+------+------+
|170.39|167.28|
|169.75|190.96|
+------+------+
only showing top 2 rows
```

In [9]:

```
#Explore features and first five rows
cw.take(5)
```

Out[9]:

```
[Row(f1=170.39, f2=167.28, f3=143.44, f4=124.67, f5=139.01, f6=125.83, f7=144.33, f8=151.26, f9=175.51,
 Row(f1=169.75, f2=190.96, f3=175.53, f4=138.27, f5=137.47, f6=139.23, f7=133.23, f8=130.25, f9=147.73,
 Row(f1=153.69, f2=153.68, f3=144.02, f4=158.73, f5=178.87, f6=157.04, f7=152.92, f8=147.52, f9=142.87,
 Row(f1=131.69, f2=151.56, f3=151.05, f4=134.0, f5=151.18, f6=175.53, f7=171.34, f8=159.77, f9=151.95, f
 Row(f1=162.85, f2=158.88, f3=132.27, f4=138.41, f5=143.98, f6=159.3, f7=177.26, f8=180.58, f9=159.34, f
```

In [10]:

```
#get total count of rows, there are 2048 rows in this dataset
cw.count()
```

Out[10]:

```
2048
```

In [11]:

```
#explore schema, according to apache documentation double refers to: "Represents 8-byte double-precision fl
cw.printSchema()
```

```
root
 |-- f1: double (nullable = true)
 |-- f2: double (nullable = true)
 |-- f3: double (nullable = true)
 |-- f4: double (nullable = true)
 |-- f5: double (nullable = true)
 |-- f6: double (nullable = true)
 |-- f7: double (nullable = true)
 |-- f8: double (nullable = true)
 |-- f9: double (nullable = true)
 |-- f10: double (nullable = true)
 |-- f11: double (nullable = true)
 |-- f12: double (nullable = true)
 |-- f13: double (nullable = true)
 |-- f14: double (nullable = true)
 |-- f15: double (nullable = true)
 |-- f16: double (nullable = true)
 |-- f17: double (nullable = true)
 |-- f18: double (nullable = true)
 |-- f19: double (nullable = true)
 |-- f20: double (nullable = true)
 |-- f21: double (nullable = true)
 |-- f22: double (nullable = true)
 |-- f23: double (nullable = true)
 |-- f24: double (nullable = true)
 |-- f25: double (nullable = true)
 |-- f26: double (nullable = true)
 |-- f27: double (nullable = true)
 |-- f28: double (nullable = true)
 |-- f29: double (nullable = true)
 |-- f30: double (nullable = true)
 |-- f31: double (nullable = true)
 |-- f32: double (nullable = true)
 |-- f33: double (nullable = true)
 |-- f34: double (nullable = true)
 |-- f35: double (nullable = true)
 |-- f36: double (nullable = true)
 |-- f37: double (nullable = true)
```

```
 |-- f38: double (nullable = true)
 |-- f39: double (nullable = true)
 |-- f40: double (nullable = true)
 |-- f41: double (nullable = true)
 |-- f42: double (nullable = true)
 |-- f43: double (nullable = true)
 |-- f44: double (nullable = true)
 |-- f45: double (nullable = true)
 |-- f46: double (nullable = true)
 |-- f47: double (nullable = true)
 |-- f48: double (nullable = true)
 |-- f49: double (nullable = true)
 |-- f50: double (nullable = true)
 |-- f51: double (nullable = true)
 |-- f52: double (nullable = true)
 |-- f53: double (nullable = true)
 |-- f54: double (nullable = true)
 |-- f55: double (nullable = true)
 |-- f56: double (nullable = true)
 |-- f57: double (nullable = true)
 |-- f58: double (nullable = true)
 |-- f59: double (nullable = true)
 |-- f60: double (nullable = true)
 |-- f61: double (nullable = true)
 |-- f62: double (nullable = true)
 |-- f63: double (nullable = true)
 |-- f64: double (nullable = true)
 |-- f65: double (nullable = true)
 |-- f66: double (nullable = true)
 |-- f67: double (nullable = true)
 |-- label: integer (nullable = true)
```

In [12]:

```
#get count of columns, there are 68 columns, in the original set there were 64 plus a label,
#it is likely there are three duplicate columns
{len(cw.columns)}
```

Out[12]:

```
{68}
```

In [13]:

```
#another way to look at dataframe dimensions
def spark_shape(data):
    return (data.count(), len(data.columns))
cw_shape = spark_shape(cw)
cw_shape
```

Out[13]:

```
(2048, 68)
```

In [14]:

```
#further exploration of features and rows
cw.show(20)
```

```
+------+------+------+------+------+------+------+------+------+------+------+------+------+------+--
|    f1|    f2|    f3|    f4|    f5|    f6|    f7|    f8|    f9|   f10|   f11|   f12|   f13|   f14|
+------+------+------+------+------+------+------+------+------+------+------+------+------+------+--
|170.39|167.28|143.44|124.67|139.01|125.83|144.33|151.26|175.51|171.31| 161.9|146.92| 141.8|140.91| 1
|169.75|190.96|175.53|138.27|137.47|139.23|133.23|130.25|147.73|163.93|167.36|171.52|155.54|139.34|15
|153.69|153.68|144.02|158.73|178.87|157.04|152.92|147.52|142.87|165.26|160.39|137.86|149.62|153.43| 1
|131.69|151.56|151.05| 134.0|151.18|175.53|171.34|159.77|151.95| 146.1|148.53|140.28|138.16|145.44| 1
|162.85|158.88|132.27|138.41|143.98| 159.3|177.26|180.58|159.34|164.66|138.04|132.76|157.88|165.58|17
|132.05|149.12|165.08|170.62|162.19| 157.1|145.86|149.52|162.84| 149.5|138.86|140.41|156.82|171.41|15
|153.59|142.25|157.33|156.08|149.33|162.97|150.25|146.47|145.99|137.82| 152.9|161.64|150.23| 170.7|18
|167.68|153.49|149.19|148.71|166.03|167.04|153.06|157.48|133.57|143.66|167.27|172.45| 179.8|169.15| 1
|136.48|130.02|131.72|152.04|163.03|172.93|170.11| 165.2|166.41|120.67|119.02|135.76|147.52|164.59|17
|145.96|140.31|126.34|113.12|118.66|140.33| 139.9|139.51| 168.7|149.54|149.38|147.88|129.45|143.81|13
|131.08| 123.3|147.32|150.99|150.73|148.27|153.97|157.52|150.63| 128.4|130.98| 150.6|151.69|146.12|14
| 170.6|147.29|151.48|149.29| 137.6|162.15|171.54|131.15|165.84|141.37|143.72|138.37| 129.1|160.41|17
|139.11|131.09|135.86|138.26|139.32|129.36|127.15|128.16|140.02|151.06|168.77|156.14|128.71|110.57|11
|153.41|173.65|168.81|143.56|130.35|128.61|138.26|132.69| 142.4|157.31|147.23|164.19|175.49| 153.2|15
|118.62|141.01|152.81|168.03|141.41|124.15|138.97|136.46|111.73|147.31| 160.7|160.62| 153.6|132.28| 1
|143.01| 150.6|160.42|160.71| 164.4|152.44|153.82|142.38|121.96|134.68|145.74|164.03|181.31| 169.5|15
|122.28|121.07|136.34|164.09|171.49| 167.5| 153.3|135.84|152.38|127.52|118.93|129.54| 143.0|158.57|15
| 144.6|158.24|156.54|130.02| 144.2|146.98|144.31|137.42|151.11|164.84|160.35|147.28|157.32| 169.6|17
|105.11|104.67|117.24|132.64|120.32|119.85|132.25|133.11|124.45|128.66|125.11|150.99|144.23|136.57|14
|144.26| 132.2|140.19|161.19|189.46|157.34|103.98|120.33| 118.7|128.71|144.17|149.85|151.25|149.96| 1
+------+------+------+------+------+------+------+------+------+------+------+------+------+------+--
only showing top 20 rows
```

In [15]:

```
#check for nan values (none in any column), there are no nan values
cw.select([count(when(isnan(c), c)).alias(c) for c in cw.columns]).show()
```

```
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---
| f1| f2| f3| f4| f5| f6| f7| f8| f9|f10|f11|f12|f13|f14|f15|f16|f17|f18|f19|f20|f21|f22|f23|f24|f25|f26
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---
|  0|  0|  0|  0|  0|  0|  0|  0|  0|  0|  0|  0|  0|  0|  0|  0|  0|  0|  0|  0|  0|  0|  0|  0|  0|  0
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---
```

In [16]:
```
#check data types
cw.dtypes
```

Out[16]:
```
[('f1', 'double'),
 ('f2', 'double'),
 ('f3', 'double'),
 ('f4', 'double'),
 ('f5', 'double'),
 ('f6', 'double'),
 ('f7', 'double'),
 ('f8', 'double'),
 ('f9', 'double'),
 ('f10', 'double'),
 ('f11', 'double'),
 ('f12', 'double'),
 ('f13', 'double'),
 ('f14', 'double'),
 ('f15', 'double'),
 ('f16', 'double'),
 ('f17', 'double'),
 ('f18', 'double'),
 ('f19', 'double'),
 ('f20', 'double'),
 ('f21', 'double'),
 ('f22', 'double'),
 ('f23', 'double'),
 ('f24', 'double'),
 ('f25', 'double'),
 ('f26', 'double'),
 ('f27', 'double'),
 ('f28', 'double'),
 ('f29', 'double'),
 ('f30', 'double'),
 ('f31', 'double'),
 ('f32', 'double'),
 ('f33', 'double'),
 ('f34', 'double'),
 ('f35', 'double'),
 ('f36', 'double'),
 ('f37', 'double'),
 ('f38', 'double'),
 ('f39', 'double'),
 ('f40', 'double'),
```

```
 ('f41', 'double'),
 ('f42', 'double'),
 ('f43', 'double'),
 ('f44', 'double'),
 ('f45', 'double'),
 ('f46', 'double'),
 ('f47', 'double'),
 ('f48', 'double'),
 ('f49', 'double'),
 ('f50', 'double'),
 ('f51', 'double'),
 ('f52', 'double'),
 ('f53', 'double'),
 ('f54', 'double'),
 ('f55', 'double'),
 ('f56', 'double'),
 ('f57', 'double'),
 ('f58', 'double'),
 ('f59', 'double'),
 ('f60', 'double'),
 ('f61', 'double'),
 ('f62', 'double'),
 ('f63', 'double'),
 ('f64', 'double'),
 ('f65', 'double'),
 ('f66', 'double'),
 ('f67', 'double'),
 ('label', 'int')]
```

In [0]:

```
#create table for statistics (count, mean, stddev, min, max)
cw_stats = cw.describe()
```

In [18]:

```
#explore statistics
cw.describe().show()
```

```
+-------+------------------+------------------+------------------+------------------+------------------+-
|summary|                f1|                f2|                f3|                f4|                f5|
+-------+------------------+------------------+------------------+------------------+------------------+-
|  count|              2048|              2048|              2048|              2048|              2048|
|   mean|125.56923144531228| 125.3568066406251|125.42204101562524|125.71933251953112|126.0050551757814|1
| stddev| 33.29273147272022|32.822212055580586| 32.64300489797472| 32.96603101800241|33.52624745933872|
|    min|            47.124|            47.262|            48.485|            49.323|            47.077|
|    max|            210.65|             210.2|            212.93|             211.0|             213.1|
+-------+------------------+------------------+------------------+------------------+------------------+-
```

In [19]:

```
#explore cw_stats dimensions
print(spark_shape(cw_stats))
```

```
(5, 69)
```

In [20]:
```
cw_stats.printSchema()
```

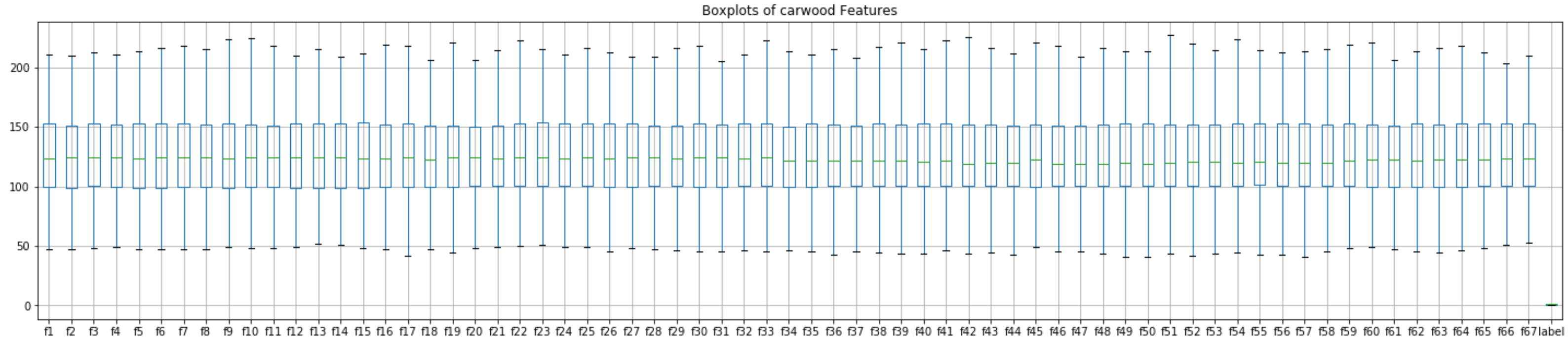
```
root
 |-- summary: string (nullable = true)
 |-- f1: string (nullable = true)
 |-- f2: string (nullable = true)
 |-- f3: string (nullable = true)
 |-- f4: string (nullable = true)
 |-- f5: string (nullable = true)
 |-- f6: string (nullable = true)
 |-- f7: string (nullable = true)
 |-- f8: string (nullable = true)
 |-- f9: string (nullable = true)
 |-- f10: string (nullable = true)
 |-- f11: string (nullable = true)
 |-- f12: string (nullable = true)
 |-- f13: string (nullable = true)
 |-- f14: string (nullable = true)
 |-- f15: string (nullable = true)
 |-- f16: string (nullable = true)
 |-- f17: string (nullable = true)
 |-- f18: string (nullable = true)
 |-- f19: string (nullable = true)
 |-- f20: string (nullable = true)
 |-- f21: string (nullable = true)
 |-- f22: string (nullable = true)
 |-- f23: string (nullable = true)
 |-- f24: string (nullable = true)
 |-- f25: string (nullable = true)
 |-- f26: string (nullable = true)
 |-- f27: string (nullable = true)
 |-- f28: string (nullable = true)
 |-- f29: string (nullable = true)
 |-- f30: string (nullable = true)
 |-- f31: string (nullable = true)
 |-- f32: string (nullable = true)
 |-- f33: string (nullable = true)
 |-- f34: string (nullable = true)
 |-- f35: string (nullable = true)
 |-- f36: string (nullable = true)
 |-- f37: string (nullable = true)
 |-- f38: string (nullable = true)
 |-- f39: string (nullable = true)
```

```
 |-- f40: string (nullable = true)
 |-- f41: string (nullable = true)
 |-- f42: string (nullable = true)
 |-- f43: string (nullable = true)
 |-- f44: string (nullable = true)
 |-- f45: string (nullable = true)
 |-- f46: string (nullable = true)
 |-- f47: string (nullable = true)
 |-- f48: string (nullable = true)
 |-- f49: string (nullable = true)
 |-- f50: string (nullable = true)
 |-- f51: string (nullable = true)
 |-- f52: string (nullable = true)
 |-- f53: string (nullable = true)
 |-- f54: string (nullable = true)
 |-- f55: string (nullable = true)
 |-- f56: string (nullable = true)
 |-- f57: string (nullable = true)
 |-- f58: string (nullable = true)
 |-- f59: string (nullable = true)
 |-- f60: string (nullable = true)
 |-- f61: string (nullable = true)
 |-- f62: string (nullable = true)
 |-- f63: string (nullable = true)
 |-- f64: string (nullable = true)
 |-- f65: string (nullable = true)
 |-- f66: string (nullable = true)
 |-- f67: string (nullable = true)
 |-- label: string (nullable = true)
```

In [0]:

```
cw_pd = cw.toPandas()
```

In [36]:
```
#it is interesting that there is no indication of outliers in this dataset
cw_pd.boxplot(figsize=(25,5))
plt.title('Boxplots of carwood Features')
plt.show()
```



In [31]:

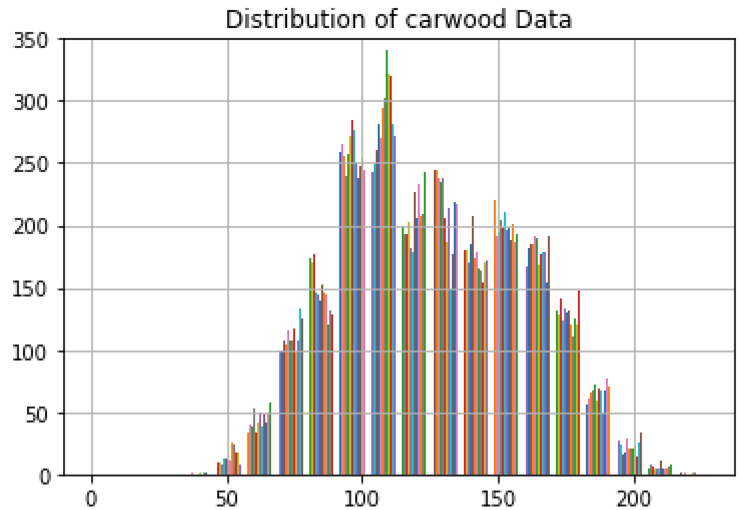
```
#package to create histogram in pyspark
!pip install pyspark_dist_explore
from pyspark_dist_explore import hist
```

```
Collecting pyspark_dist_explore
  Downloading https://files.pythonhosted.org/packages/3c/33/2b6c29265413f2b56516caf02b8befbb6a79a1a3516c
Requirement already satisfied: pandas in /usr/local/lib/python3.6/dist-packages (from pyspark_dist_explo
Requirement already satisfied: numpy in /usr/local/lib/python3.6/dist-packages (from pyspark_dist_explor
Requirement already satisfied: matplotlib in /usr/local/lib/python3.6/dist-packages (from pyspark_dist_e
Requirement already satisfied: scipy in /usr/local/lib/python3.6/dist-packages (from pyspark_dist_explor
Requirement already satisfied: pytz>=2017.2 in /usr/local/lib/python3.6/dist-packages (from pandas->pysp
Requirement already satisfied: python-dateutil>=2.6.1 in /usr/local/lib/python3.6/dist-packages (from pa
Requirement already satisfied: cycler>=0.10 in /usr/local/lib/python3.6/dist-packages (from matplotlib->
Requirement already satisfied: kiwisolver>=1.0.1 in /usr/local/lib/python3.6/dist-packages (from matplot
Requirement already satisfied: pyparsing!=2.0.4,!=2.1.2,!=2.1.6,>=2.0.1 in /usr/local/lib/python3.6/dist
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.6/dist-packages (from python-dateutil>
Requirement already satisfied: setuptools in /usr/local/lib/python3.6/dist-packages (from kiwisolver>=1
Installing collected packages: pyspark-dist-explore
Successfully installed pyspark-dist-explore-0.1.8
```

In [38]:

```
#the overall distribution of the dataset appears relatively normal but somewhat multimodal
fig = plt.figure(figsize=(8, 4))
fig, ax = plt.subplots()
ax.set_ylim([0,350])
hist(ax, cw, bins = 20)
plt.title('Distribution of carwood Data')
plt.grid()
plt.show()
```

```
<Figure size 576x288 with 0 Axes>
```



In [39]:

```
cw_pd.describe()
```

Out[39]:

	f1	f2	f3	f4	f5	f6	f7	f8	f9
count	2048.000000	2048.000000	2048.000000	2048.000000	2048.000000	2048.000000	2048.000000	2048.000000	2048.000000
mean	125.569231	125.356807	125.422041	125.719333	126.005055	126.084465	125.885123	125.783833	125.704766
std	33.292731	32.822212	32.643005	32.966031	33.526247	33.589233	33.220224	33.214697	33.548514
min	47.124000	47.262000	48.485000	49.323000	47.077000	47.365000	47.063000	47.546000	49.302000
25%	99.490000	99.095500	100.217500	99.784750	99.094250	98.990500	99.144750	99.745750	98.876750
50%	123.430000	124.160000	123.970000	124.460000	123.735000	124.275000	124.465000	124.390000	123.425000
75%	153.017500	151.252500	152.505000	152.347500	152.677500	153.165000	153.202500	152.085000	152.965000
max	210.650000	210.200000	212.930000	211.000000	213.100000	215.900000	218.090000	215.430000	223.880000

In [0]:

```
cw_stats_pd = cw_pd.describe()
```

In [41]:

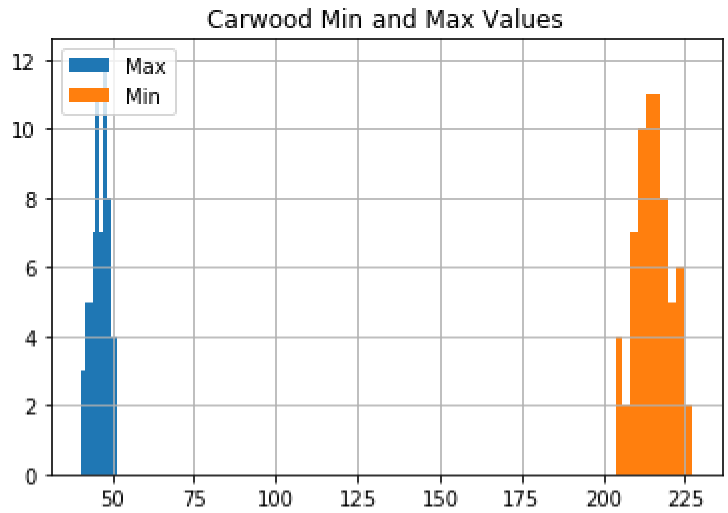
```
#overall mean of the dataset is 123.47 and the overall median is 120.70. This is indicative of a relatively
# distribution. The range of the dataset is 166.09
#with a standard deviation of 32.93. The minimum value is 45.67 and the max value is 211.76.
cw_stats_mean = cw_stats_pd.mean(axis=1)
cw_stats_mean
```

Out[41]:

```
count    2048.000000
mean      123.474950
std        32.922866
min        45.673441
25%        98.535467
50%       120.702868
75%       149.982096
max       211.762794
dtype: float64
```

In [43]:
```
cw_stats_Max = cw_stats_pd.iloc[3,:66]
cw_stats_Min = cw_stats_pd.iloc[7,:66]

plt.hist(cw_stats_Max, label='Max')
plt.hist(cw_stats_Min, label='Min')
plt.legend(loc='upper left')
plt.title('Carwood Min and Max Values')
plt.grid()
plt.show()
```



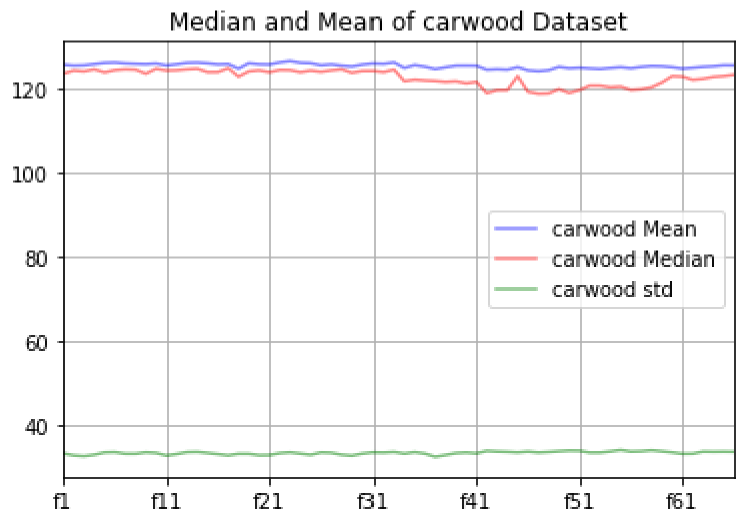
In [44]:

```
cw_stats_mean = cw_stats_pd.iloc[1,:66]
cw_stats_mean.plot(title = 'Median and Mean of carwood Dataset', legend = True, label = 'carwood Mean', col

cw_stats_median = cw_stats_pd.iloc[5,:66]
cw_stats_median.plot(legend = True, label = 'carwood Median', color = 'red', alpha = .5)

cw_stats_std = cw_stats_pd.iloc[2, :66]
cw_stats_std.plot(legend = True, label = 'carwood std', color = 'green', alpha = .5)

plt.grid()
plt.show()
```



Step 4 - For purposes of normalization/standardization discussion exploring some table statistics further

In [0]:

```
cols=cw_stats.columns[1:]
```

All features have a similar kurtosis of between negative .68 and .91. This indicates that distributions of features are platykurtic having lighter tails and heavier peaks. Given that these values are between 0 and 1 and there is very little range between kurtosis values one can inference that there is a reasonably normal distribution among the features.

Likewise, skewness values are between .11 and .21 indicating very little skewness in each feature (acceptable levels being less than 1 or greater than -1.) Again one could inference based on these values and the relative small range of skewness values between features that there is a reasonably normal distribution among the features.

In [46]:

```
#exploring skewness and kurtosis for each feature
from pyspark.sql.functions import col, kurtosis, skewness
for x in cw:
  cw.select(skewness(x),kurtosis(x)).show()
```

```
+------------------+------------------+
|      skewness(f1)|      kurtosis(f1)|
+------------------+------------------+
|0.1807989240714351|-0.827452318873823|
+------------------+------------------+

+------------------+-------------------+
|      skewness(f2)|       kurtosis(f2)|
+------------------+-------------------+
|0.1596601564377789|-0.7984791679400449|
+------------------+-------------------+

+-------------------+-------------------+
|       skewness(f3)|       kurtosis(f3)|
+-------------------+-------------------+
|0.12976889499128577|-0.8662434667307477|
+-------------------+-------------------+

+------------------+------------------+
|      skewness(f4)|      kurtosis(f4)|
+------------------+------------------+
|0.1276872593690517|-0.871604990594411|
+------------------+------------------+

+-------------------+-------------------+
|       skewness(f5)|       kurtosis(f5)|
+-------------------+-------------------+
|0.16468711882060505|-0.8387916170258198|
+-------------------+-------------------+

+-------------------+-------------------+
|       skewness(f6)|       kurtosis(f6)|
+-------------------+-------------------+
|0.16182670319181483|-0.8512719461220066|
+-------------------+-------------------+

+-------------------+-------------------+
|       skewness(f7)|       kurtosis(f7)|
```

```
+------------------+------------------+
|0.12946155723308383|-0.8882710941930139|
+------------------+------------------+

+------------------+------------------+
|       skewness(f8)|       kurtosis(f8)|
+------------------+------------------+
|0.17517859830586263|-0.8087163962071791|
+------------------+------------------+

+------------------+-----------------+
|       skewness(f9)|      kurtosis(f9)|
+------------------+-----------------+
|0.17223769660829044|-0.817272289220675|
+------------------+-----------------+

+------------------+------------------+
|      skewness(f10)|      kurtosis(f10)|
+------------------+------------------+
|0.16716953000631848|-0.8045881463753761|
+------------------+------------------+

+------------------+------------------+
|      skewness(f11)|      kurtosis(f11)|
+------------------+------------------+
|0.15730709709176205|-0.7682856575026391|
+------------------+------------------+

+-----------------+------------------+
|     skewness(f12)|      kurtosis(f12)|
+-----------------+------------------+
|0.1561556101974387|-0.8435537333979677|
+-----------------+------------------+

+------------------+------------------+
|      skewness(f13)|      kurtosis(f13)|
+------------------+------------------+
|0.18080937972427558|-0.8168214905255224|
+------------------+------------------+

+------------------+------------------+
|      skewness(f14)|      kurtosis(f14)|
+------------------+------------------+
```

```
|0.14916373739177768|-0.9057672259418217|
+-------------------+-------------------+

+-----------------+-------------------+
|   skewness(f15)|     kurtosis(f15)|
+-----------------+-------------------+
|0.139367377047339|-0.8951702231751204|
+-----------------+-------------------+

+-------------------+-------------------+
|      skewness(f16)|      kurtosis(f16)|
+-------------------+-------------------+
|0.15878194332614903|-0.8129809767214229|
+-------------------+-------------------+

+-------------------+-------------------+
|      skewness(f17)|      kurtosis(f17)|
+-------------------+-------------------+
|0.08096969467704704|-0.8434880670788449|
+-------------------+-------------------+

+-------------------+------------------+
|      skewness(f18)|     kurtosis(f18)|
+-------------------+------------------+
|0.14705089283419798|-0.792119921177421|
+-------------------+------------------+

+-------------------+-------------------+
|      skewness(f19)|      kurtosis(f19)|
+-------------------+-------------------+
|0.15377190206772906|-0.7615418448764015|
+-------------------+-------------------+

+-------------------+-------------------+
|      skewness(f20)|      kurtosis(f20)|
+-------------------+-------------------+
|0.15984917886355166|-0.7486018392690927|
+-------------------+-------------------+

+-------------------+-------------------+
|      skewness(f21)|      kurtosis(f21)|
+-------------------+-------------------+
|0.18043808891851493|-0.7394269115235104|
```

```
+-------------------+-------------------+

+-------------------+-------------------+
|      skewness(f22)|      kurtosis(f22)|
+-------------------+-------------------+
|0.14885491026278055|-0.8478916582518634|
+-------------------+-------------------+

+------------------+-------------------+
|     skewness(f23)|      kurtosis(f23)|
+------------------+-------------------+
|0.1510583691923656|-0.8522979611878827|
+------------------+-------------------+

+-------------------+-------------------+
|      skewness(f24)|      kurtosis(f24)|
+-------------------+-------------------+
|0.14333471242490742|-0.8290293865574552|
+-------------------+-------------------+

+-------------------+-------------------+
|      skewness(f25)|      kurtosis(f25)|
+-------------------+-------------------+
|0.12754749990192288|-0.8337611778909761|
+-------------------+-------------------+

+-------------------+-------------------+
|      skewness(f26)|      kurtosis(f26)|
+-------------------+-------------------+
|0.13427260684124723|-0.8094633399481923|
+-------------------+-------------------+

+-------------------+-------------------+
|      skewness(f27)|      kurtosis(f27)|
+-------------------+-------------------+
|0.12877399376315463|-0.8128397129102103|
+-------------------+-------------------+

+-------------------+-------------------+
|      skewness(f28)|      kurtosis(f28)|
+-------------------+-------------------+
|0.13232606249327525|-0.7767721195376485|
+-------------------+-------------------+
```

```
+-------------------+-------------------+
|      skewness(f29)|      kurtosis(f29)|
+-------------------+-------------------+
|0.11648044625953614|-0.8132026905513974|
+-------------------+-------------------+

+-------------------+-------------------+
|      skewness(f30)|      kurtosis(f30)|
+-------------------+-------------------+
|0.13924559726245198|-0.7835172507407848|
+-------------------+-------------------+

+-------------------+-------------------+
|      skewness(f31)|      kurtosis(f31)|
+-------------------+-------------------+
|0.13571037021340854|-0.8337299314802005|
+-------------------+-------------------+

+-------------------+-------------------+
|      skewness(f32)|      kurtosis(f32)|
+-------------------+-------------------+
|0.11541454920109137|-0.8838093418992417|
+-------------------+-------------------+

+-------------------+-------------------+
|      skewness(f33)|      kurtosis(f33)|
+-------------------+-------------------+
|0.13854051127589168|-0.8102657567681288|
+-------------------+-------------------+

+-------------------+-------------------+
|      skewness(f34)|      kurtosis(f34)|
+-------------------+-------------------+
|0.19403902921096533|-0.7517707279820316|
+-------------------+-------------------+

+-------------------+-------------------+
|      skewness(f35)|      kurtosis(f35)|
+-------------------+-------------------+
|0.16278521633988893|-0.8150389261974955|
+-------------------+-------------------+
```

```
+-------------------+-------------------+
|      skewness(f36)|      kurtosis(f36)|
+-------------------+-------------------+
|0.18844862651565686|-0.7196767031450757|
+-------------------+-------------------+

+------------------+-------------------+
|     skewness(f37)|      kurtosis(f37)|
+------------------+-------------------+
|0.1245130071584265|-0.7957066537955724|
+------------------+-------------------+

+-------------------+-------------------+
|      skewness(f38)|      kurtosis(f38)|
+-------------------+-------------------+
|0.15883124031869905|-0.8033608592000792|
+-------------------+-------------------+

+-------------------+-------------------+
|      skewness(f39)|      kurtosis(f39)|
+-------------------+-------------------+
|0.20157070919610703|-0.7286906479719883|
+-------------------+-------------------+

+------------------+-------------------+
|     skewness(f40)|      kurtosis(f40)|
+------------------+-------------------+
|0.1933676097023538|-0.8112312593122888|
+------------------+-------------------+

+-------------------+-------------------+
|      skewness(f41)|      kurtosis(f41)|
+-------------------+-------------------+
|0.20831883172123267|-0.7568238430815128|
+-------------------+-------------------+

+-------------------+-------------------+
|      skewness(f42)|      kurtosis(f42)|
+-------------------+-------------------+
|0.17718000605222975|-0.7197167034110863|
+-------------------+-------------------+

+-------------------+-------------------+
```

```
|      skewness(f43)|      kurtosis(f43)|
+-------------------+-------------------+
|0.14809904536847407|-0.7761165723056207|
+-------------------+-------------------+

+-------------------+-------------------+
|      skewness(f44)|      kurtosis(f44)|
+-------------------+-------------------+
|0.15653492155354062|-0.7479601248728391|
+-------------------+-------------------+

+-------------------+-------------------+
|      skewness(f45)|      kurtosis(f45)|
+-------------------+-------------------+
|0.13904700391380195|-0.7779148667130982|
+-------------------+-------------------+

+-------------------+-------------------+
|      skewness(f46)|      kurtosis(f46)|
+-------------------+-------------------+
|0.18350220816734011|-0.6861728758669066|
+-------------------+-------------------+

+-------------------+-------------------+
|      skewness(f47)|      kurtosis(f47)|
+-------------------+-------------------+
|0.16482210750454349|-0.7422183452793796|
+-------------------+-------------------+

+-------------------+-------------------+
|      skewness(f48)|      kurtosis(f48)|
+-------------------+-------------------+
|0.17065858923551433|-0.7502398541477557|
+-------------------+-------------------+

+------------------+-------------------+
|     skewness(f49)|      kurtosis(f49)|
+------------------+-------------------+
|0.1484650147929236|-0.7525191365720088|
+------------------+-------------------+

+-------------------+-------------------+
|      skewness(f50)|      kurtosis(f50)|
```

```
+-------------------+-------------------+
|0.16747666406894576|-0.7679318338678671|
+-------------------+-------------------+

+-------------------+-------------------+
|      skewness(f51)|      kurtosis(f51)|
+-------------------+-------------------+
|0.18071117283583488|-0.7081313691686102|
+-------------------+-------------------+

+------------------+-------------------+
|     skewness(f52)|      kurtosis(f52)|
+------------------+-------------------+
|0.1354035475376573|-0.7048947793416365|
+------------------+-------------------+

+-------------------+------------------+
|      skewness(f53)|     kurtosis(f53)|
+-------------------+------------------+
|0.14220282290537067|-0.731407007429707|
+-------------------+------------------+

+------------------+-------------------+
|     skewness(f54)|      kurtosis(f54)|
+------------------+-------------------+
|0.1448369752868945|-0.7843375783289859|
+------------------+-------------------+

+------------------+-------------------+
|     skewness(f55)|      kurtosis(f55)|
+------------------+-------------------+
|0.1592251349264446|-0.7478299177801624|
+------------------+-------------------+

+-------------------+-------------------+
|      skewness(f56)|      kurtosis(f56)|
+-------------------+-------------------+
|0.14366097362548216|-0.7628595747711961|
+-------------------+-------------------+

+------------------+-------------------+
|     skewness(f57)|      kurtosis(f57)|
+------------------+-------------------+
```

```
|0.1484650147929236|-0.7525191365720088|
+------------------+-------------------+

+------------------+-------------------+
|     skewness(f58)|      kurtosis(f58)|
+------------------+-------------------+
|0.1604382873027316|-0.7603351195040542|
+------------------+-------------------+

+-------------------+-------------------+
|      skewness(f59)|      kurtosis(f59)|
+-------------------+-------------------+
|0.16623552915496792|-0.6995633407649628|
+-------------------+-------------------+

+-------------------+-------------------+
|      skewness(f60)|      kurtosis(f60)|
+-------------------+-------------------+
|0.13904700391380195|-0.7779148667130982|
+-------------------+-------------------+

+-------------------+------------------+
|      skewness(f61)|     kurtosis(f61)|
+-------------------+------------------+
|0.14705089283419798|-0.792119921177421|
+-------------------+------------------+

+-------------------+-------------------+
|      skewness(f62)|      kurtosis(f62)|
+-------------------+-------------------+
|0.13676321447067705|-0.8279358477223453|
+-------------------+-------------------+

+-------------------+-------------------+
|      skewness(f63)|      kurtosis(f63)|
+-------------------+-------------------+
|0.17517009164530264|-0.7582680603073224|
+-------------------+-------------------+

+-------------------+-------------------+
|      skewness(f64)|      kurtosis(f64)|
+-------------------+-------------------+
|0.13227833595202648|-0.8216221064639089|
```

```
+------------------+------------------+

+------------------+------------------+
|     skewness(f65)|     kurtosis(f65)|
+------------------+------------------+
|0.12402360832201269|-0.8237769443075966|
+------------------+------------------+

+------------------+------------------+
|     skewness(f66)|     kurtosis(f66)|
+------------------+------------------+
|0.13364735299420766|-0.8351266259402292|
+------------------+------------------+

+-----------------+------------------+
|    skewness(f67)|     kurtosis(f67)|
+-----------------+------------------+
|0.1469029657632442|-0.7934727859986483|
+-----------------+------------------+

+-------------------+------------------+
|    skewness(label)|   kurtosis(label)|
+-------------------+------------------+
|-0.00585940014587...|-1.9999656674299287|
+-------------------+------------------+
```

Step 4 continued:

The boxplots of the features visualized below show a fairly consistant spread and median among the features. The histograms also show that the plots are fairly similar in terms of distribution. There are also no outliers in the boxplots and the range of values between the features is consistant. Some features do have a few differences in terms of the degree to which the are bimodal or multimodal. Likewise the histogram of the data overall shows a normal distribution.

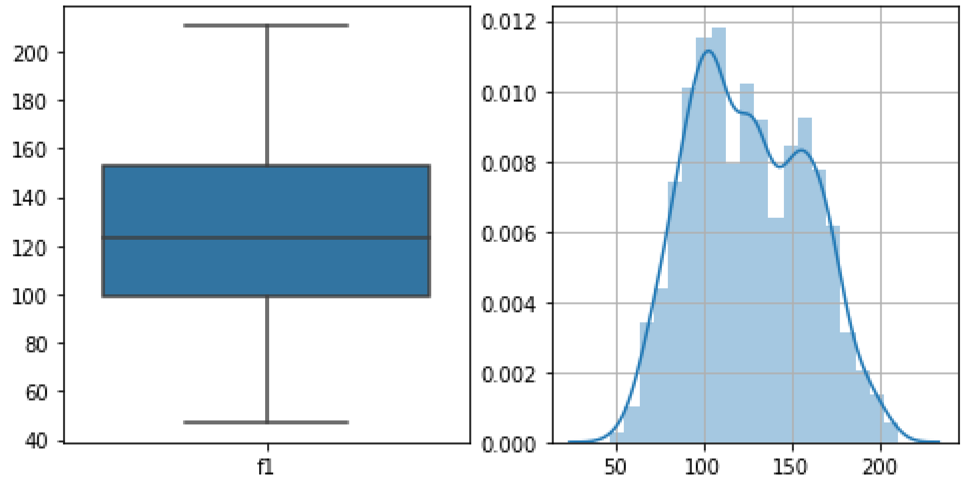
Should we use normalization or standardization? Normalization (Min-max) is used when the objective is to bring the values of a feature or features to a range of 0 to 1 through rescaling. Standardization (z-score or standard scaler), on the otherhand, also rescales the data but instead brings the values of a feature or features to have a mean of zero and a standard deviation of 1, effectively resulting in the distribution of the data being between -1 and 1.

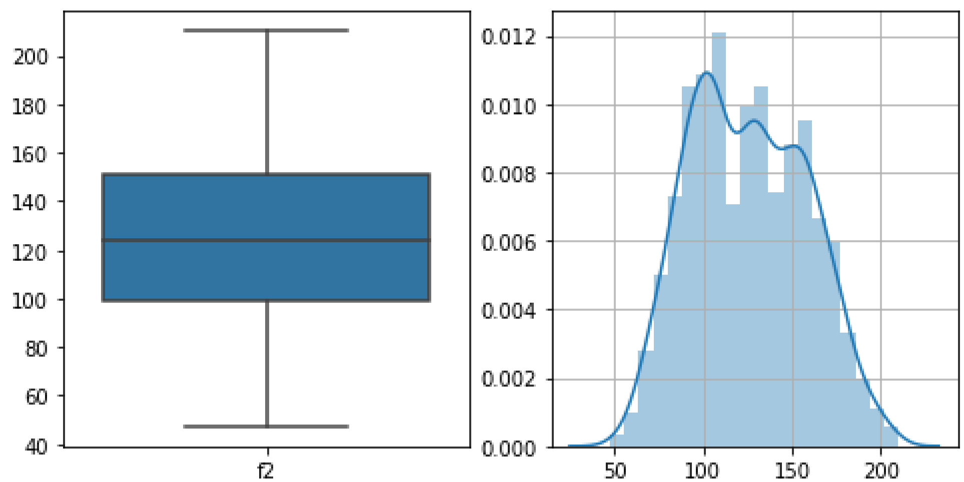
Typically, normalization or standardization are used to transform data when these types of changes to the data would correct issues that make the analysis of the data difficult to do. For example, if values are extremely different in terms of range, minimum and maximum values, then rescaling these values would make them easier to compare. Another reason to rescale data would be if there were significant outliers in the data or if variables that were to be compared in a model had different units of measure. Finally, sometimes certain machine learning algorithms (like principal compenent analysis) respond better when the data is normalized or standardized first.

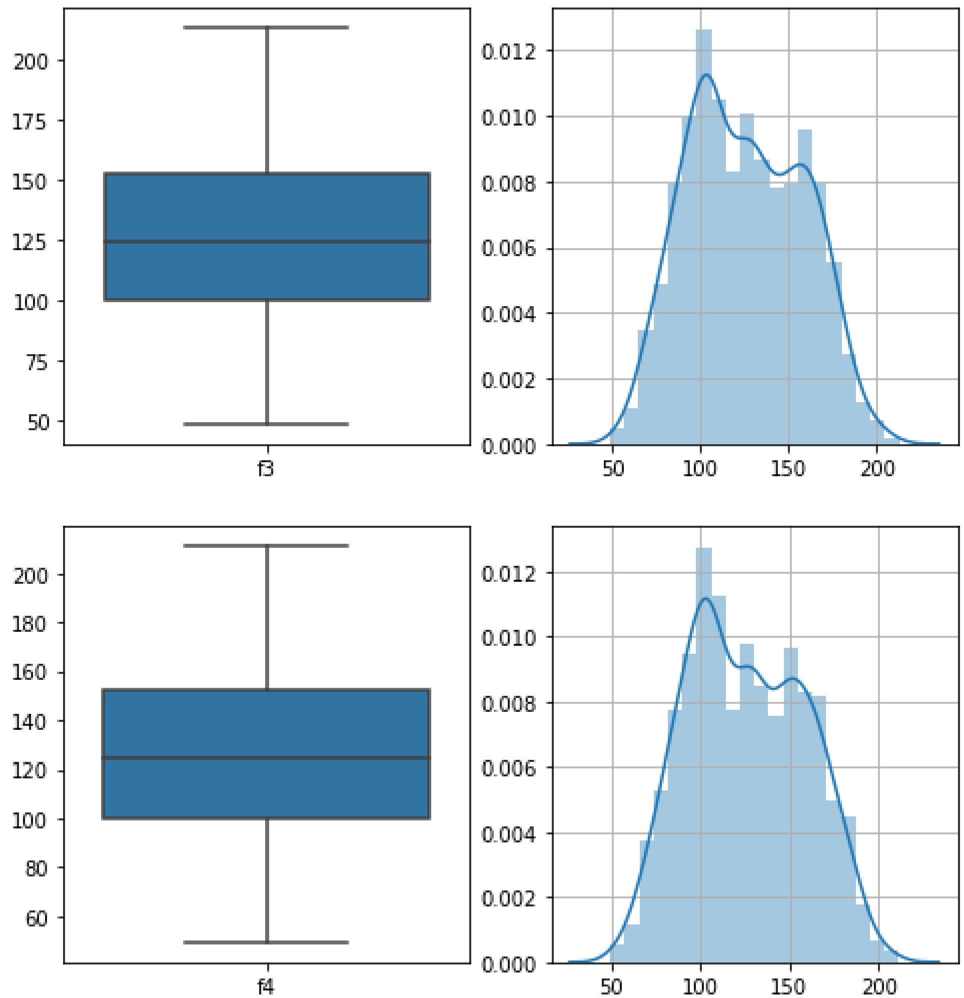
Presently, since our data has a normal distribution and no outliers it does not appear that normaization or standardization is necessary at this point. Furthermore, all of our data features represent values that had the same unit of measure and the range of the numbers is fairly compact. At this time then these techniques would not need to be applied. However, should we utilize these sets in certain machine learning algorithms it could become necesary or advisable to standardize the data set before performing the machine learning algorithm.

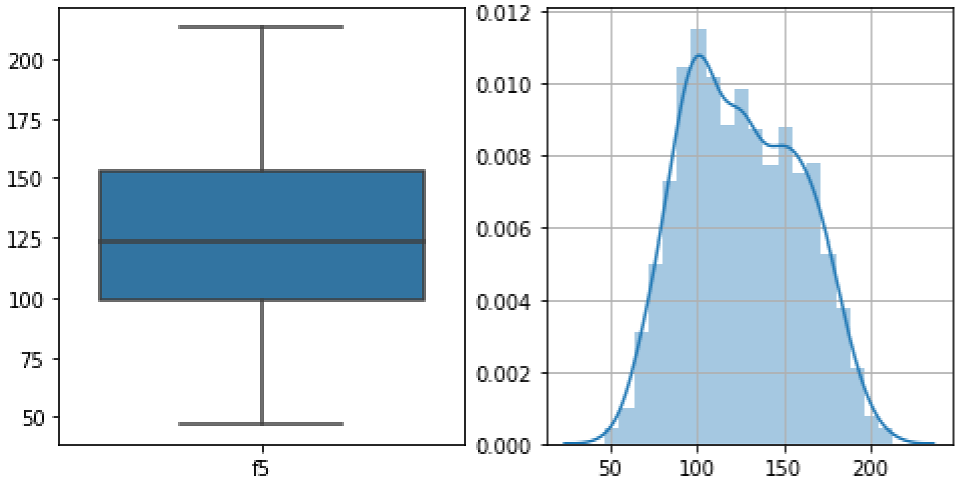
In [47]:

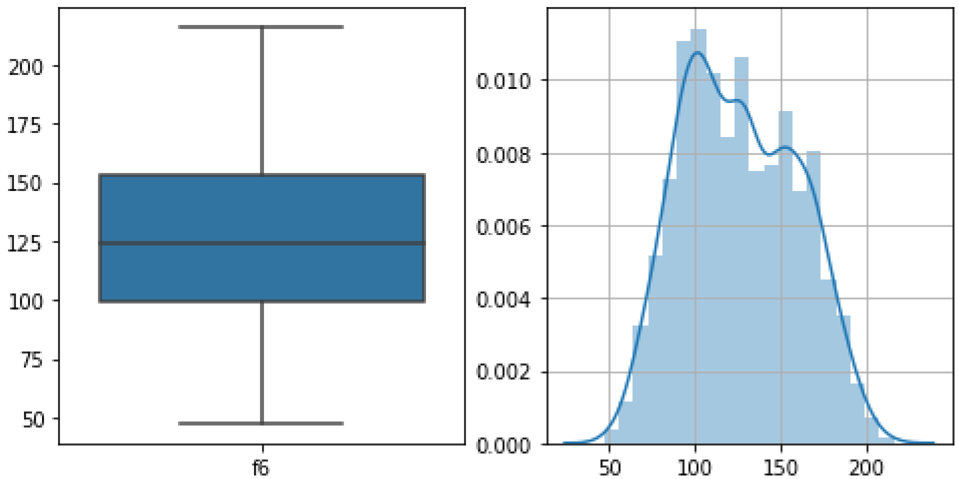
```
#using visualizations to explore similarities and differences between features:
import matplotlib.pyplot as plt
import seaborn as sns
for y in cw:
  x = cw.select(y).toPandas()
  fig = plt.figure(figsize=(8, 4))
  ax = fig.add_subplot(1, 2, 1)
  ax = sns.boxplot(data=x)
  ax = fig.add_subplot(1, 2, 2)
  ax = sns.distplot(x)
  plt.grid()
  plt.show()
```



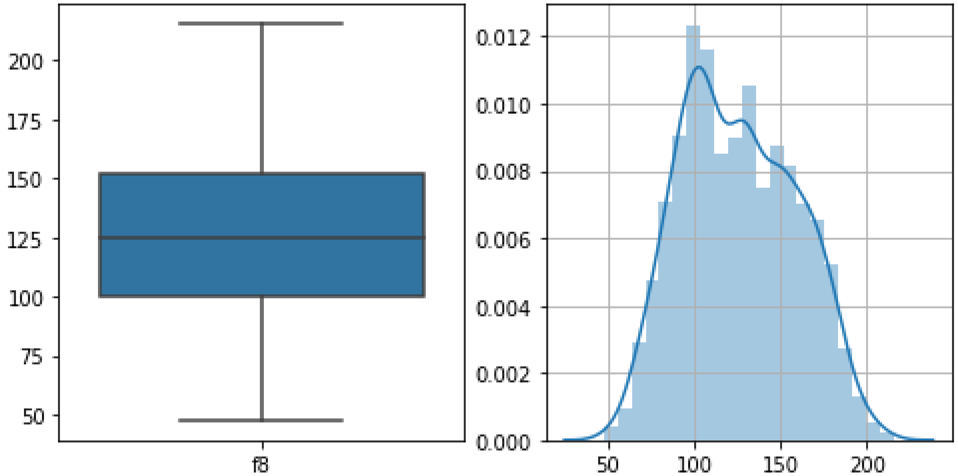


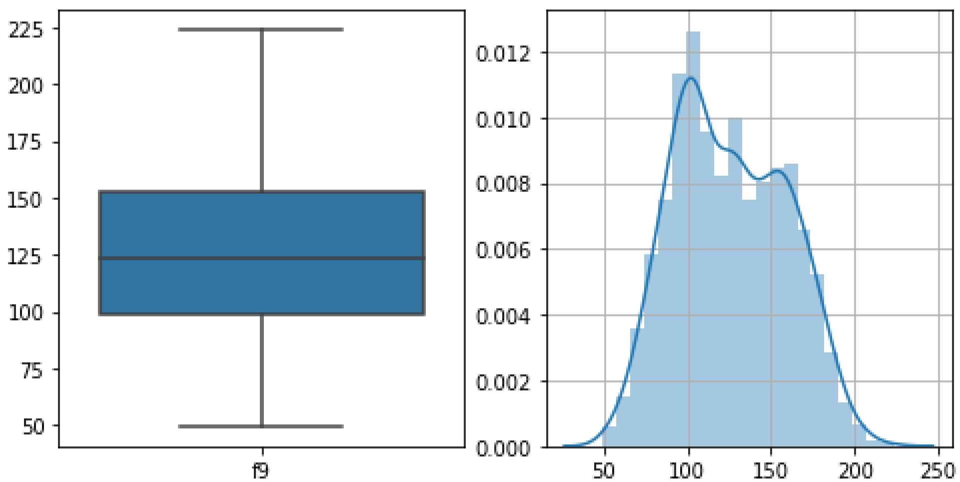

f3
f4

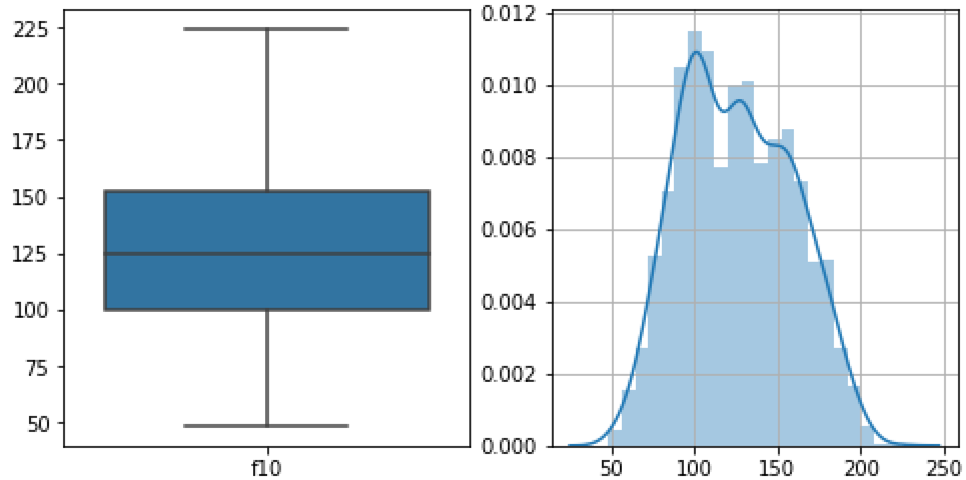

200
175
150
125
100
75
50
f5
0.012
0.010
0.008
0.006
0.004
0.002
0.000
50
100
150
200

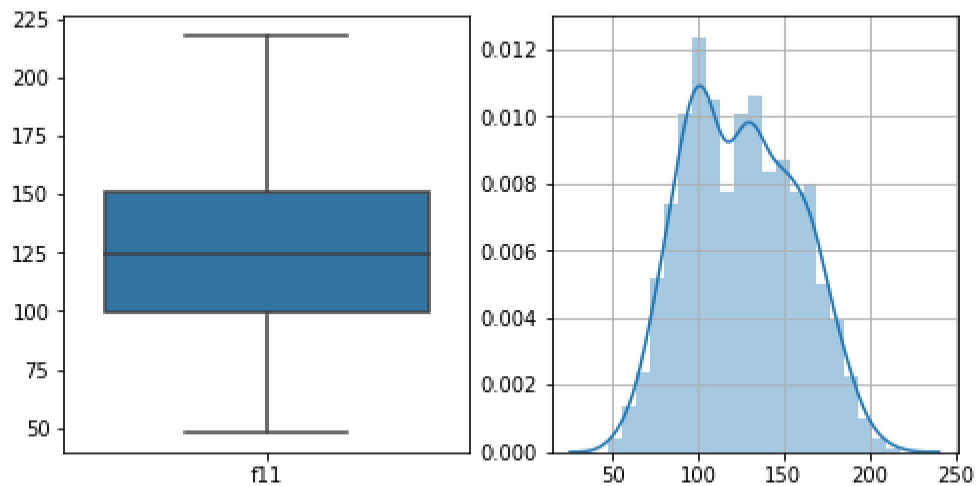

200
175
150
125
100
75
50
f6
0.010
0.008
0.006
0.004
0.002
0.000
50
100
150
200

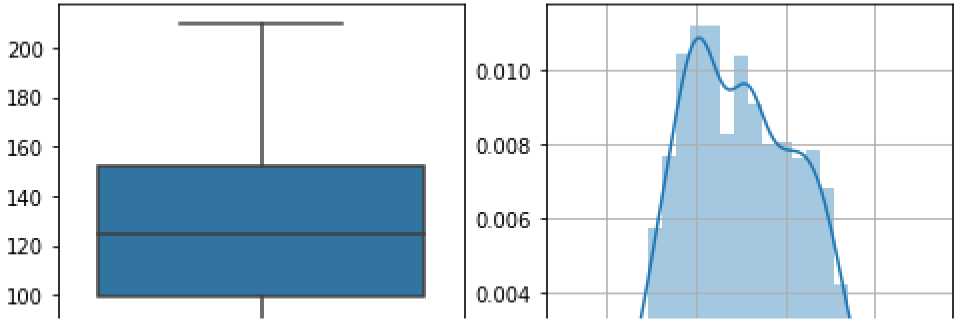

225

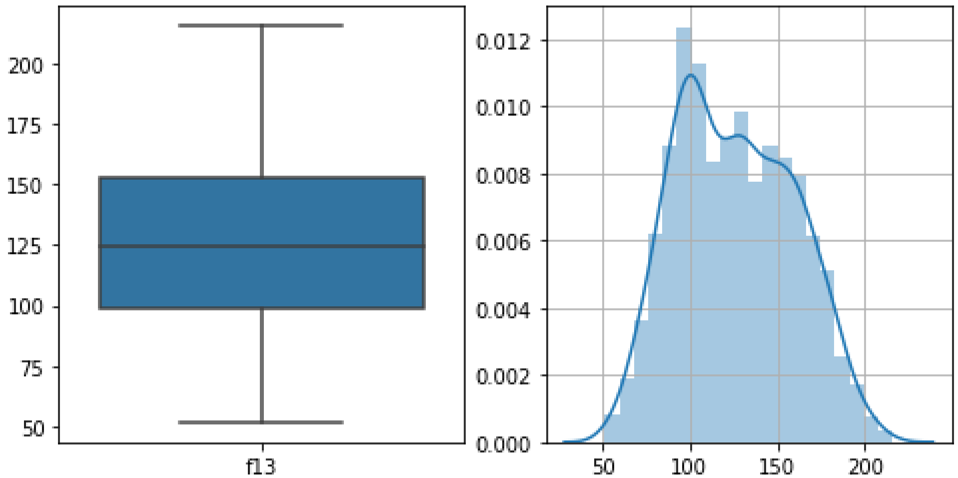

200
175
150
125
100
75
50
f8
0.012
0.010
0.008
0.006
0.004
0.002
0.000
50
100
150
200

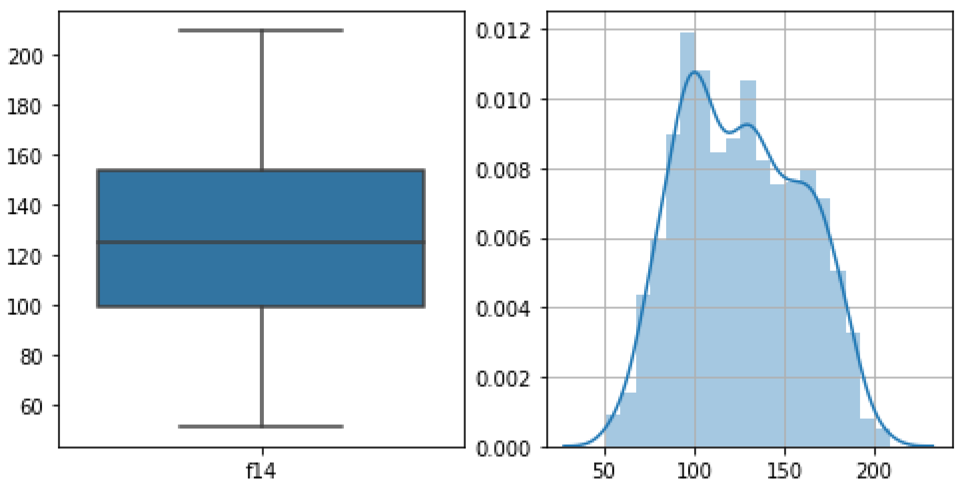

225
200
175
150
125
100
75
50
f9
0.012
0.010
0.008
0.006
0.004
0.002
0.000
50
100
150
200
250

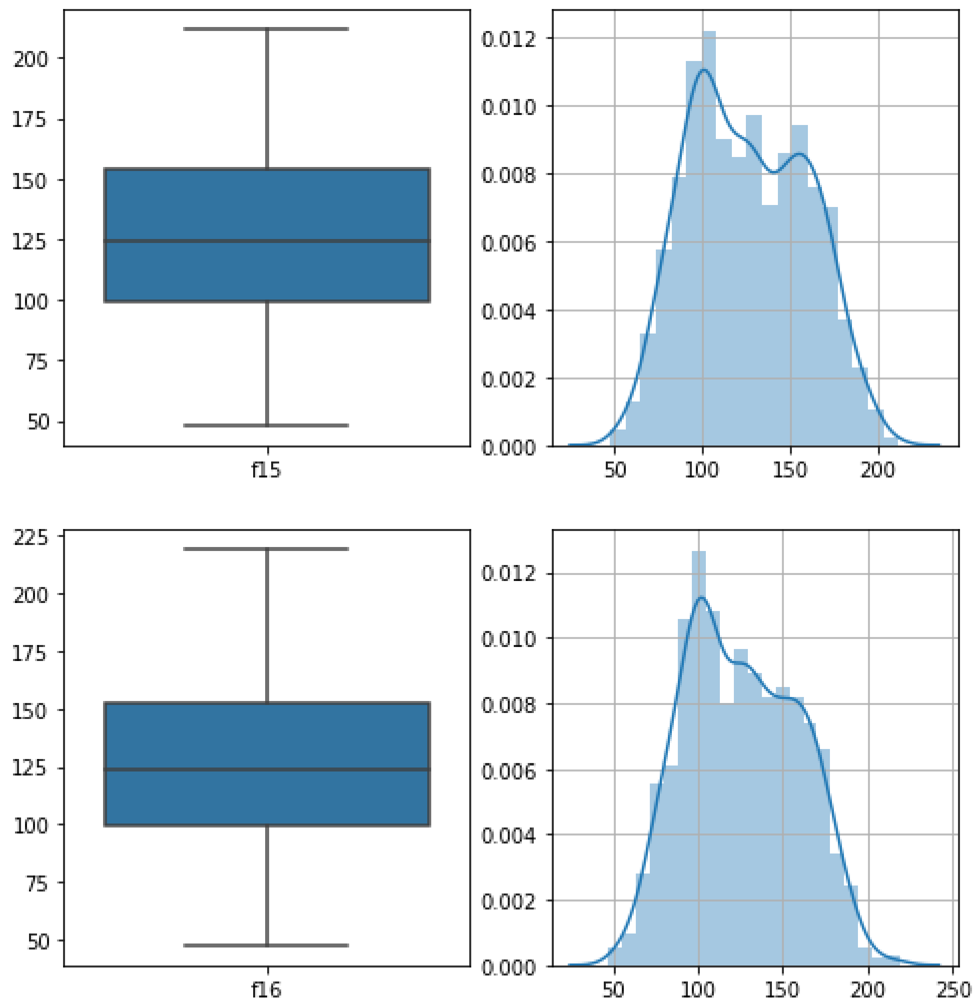

f10

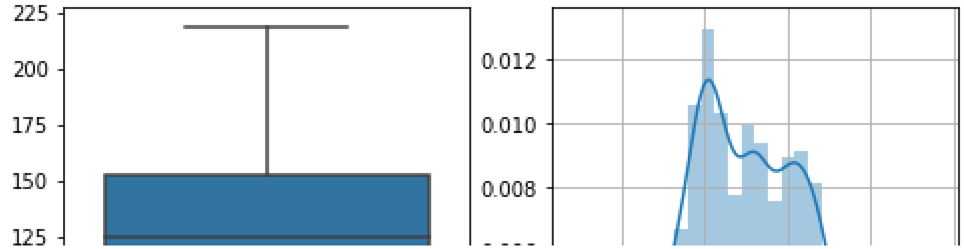

f11

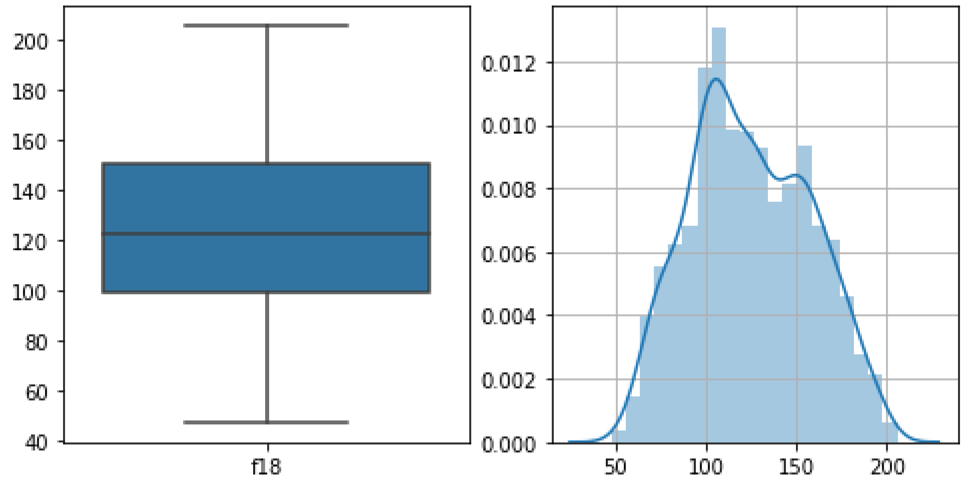


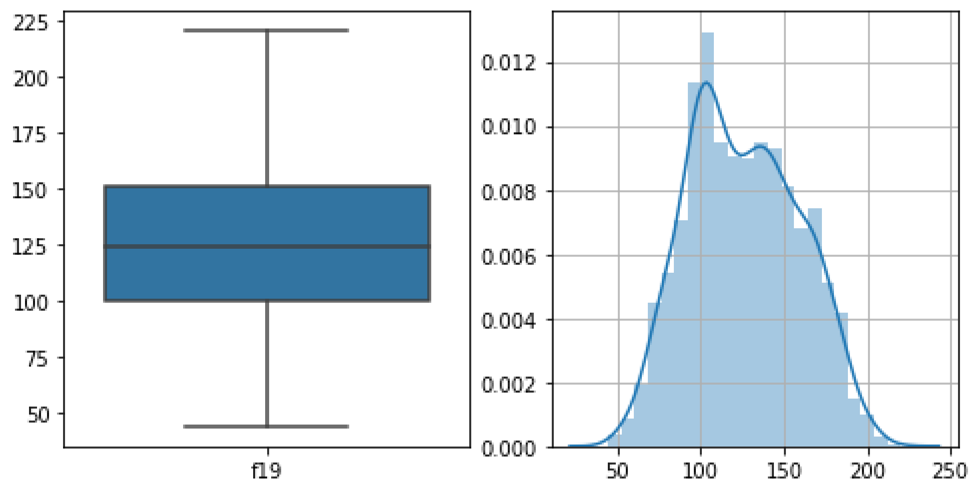

f13

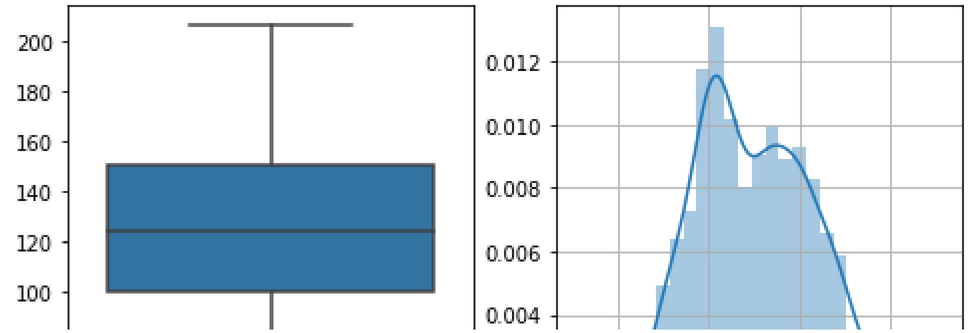

f14

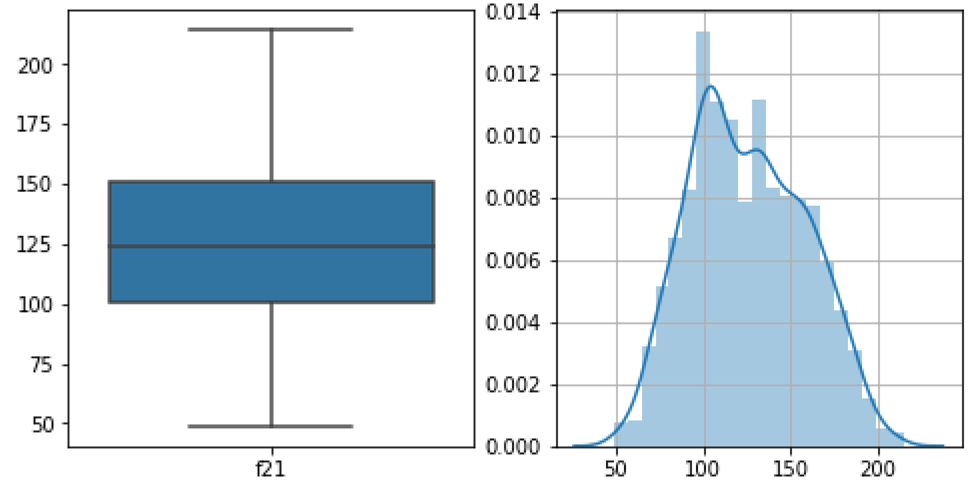

f15
f16

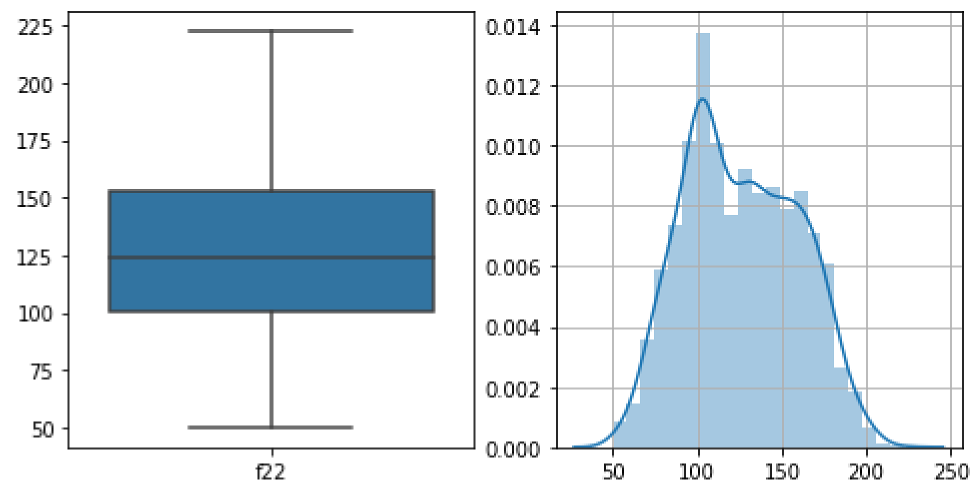

225
200
175
150
125
0.012
0.010
0.008

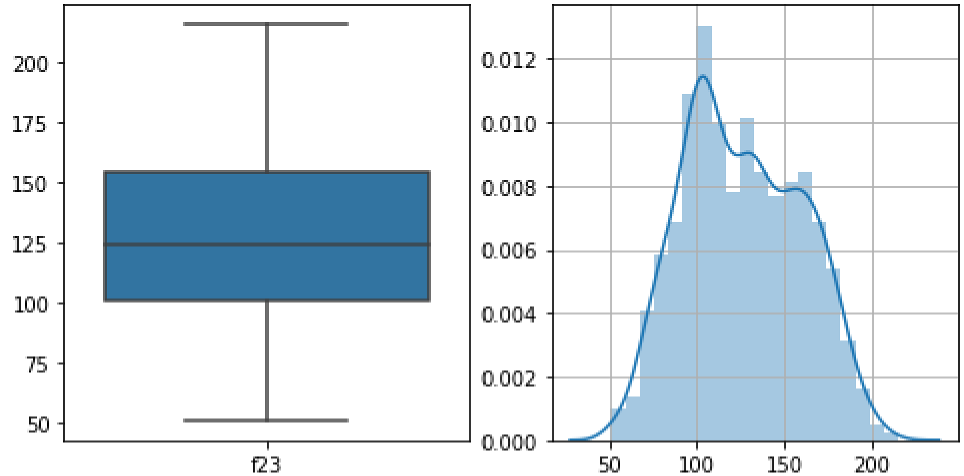

200
180
160
140
120
100
80
60
40
f18
0.012
0.010
0.008
0.006
0.004
0.002
0.000
50
100
150
200


225
200
175
150
125
100
75
50
f19
0.012
0.010
0.008
0.006
0.004
0.002
0.000
50
100
150
200
250

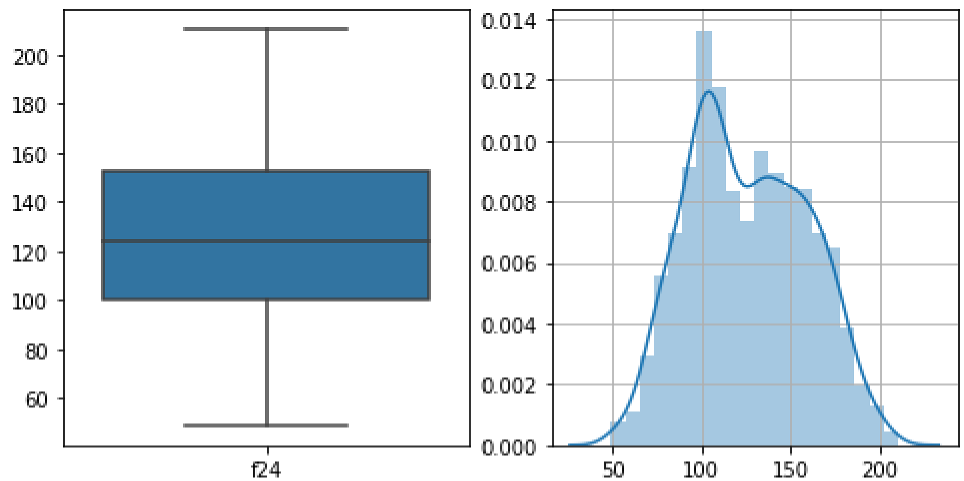



f21

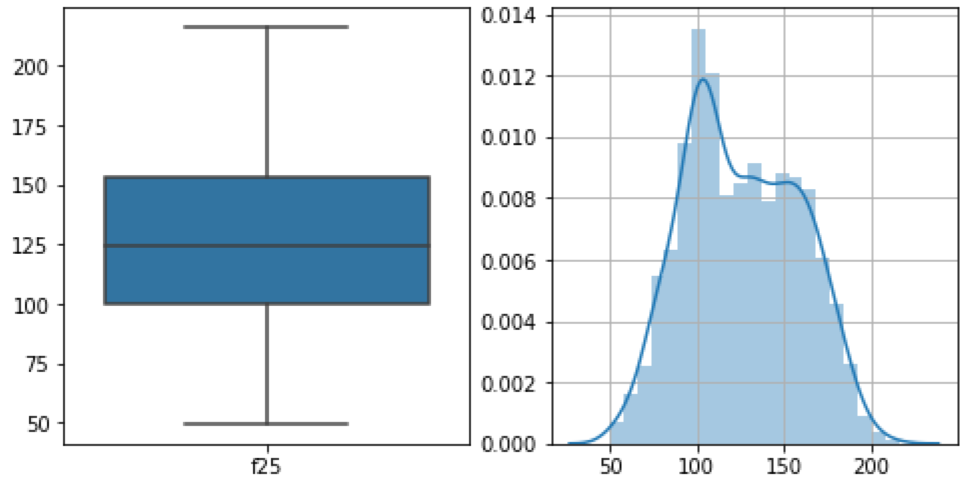

f22

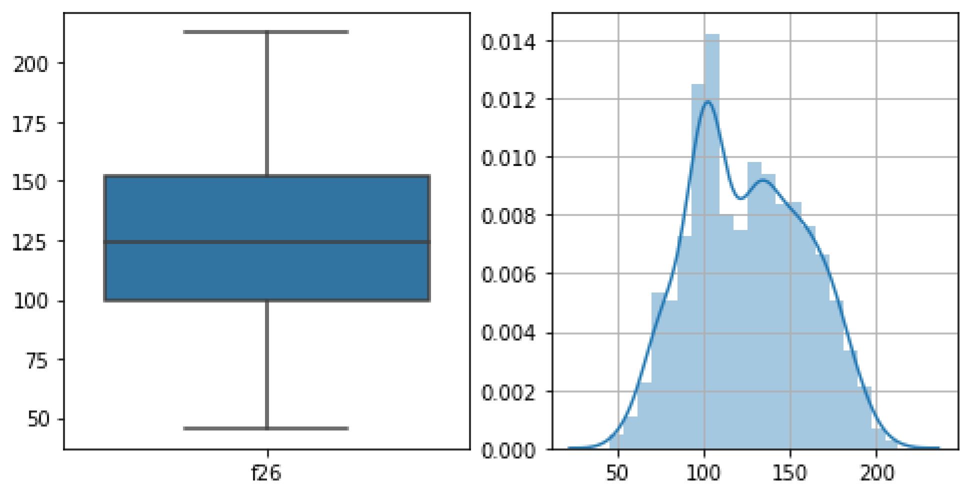


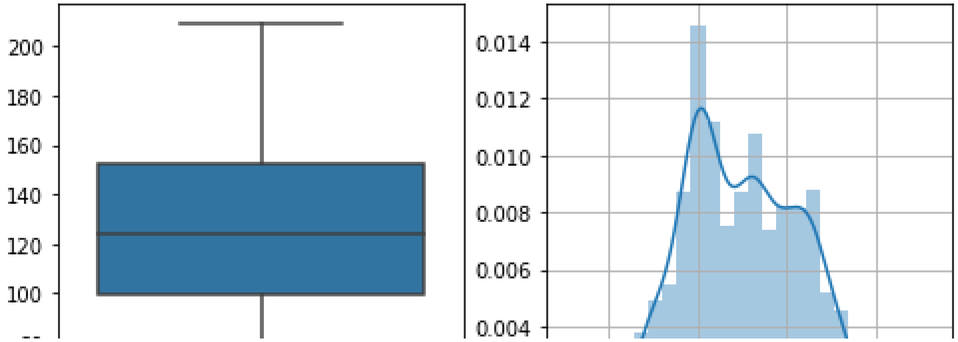
200
175
150
125
100
75
50
f23
0.012
0.010
0.008
0.006
0.004
0.002
0.000
50
100
150
200

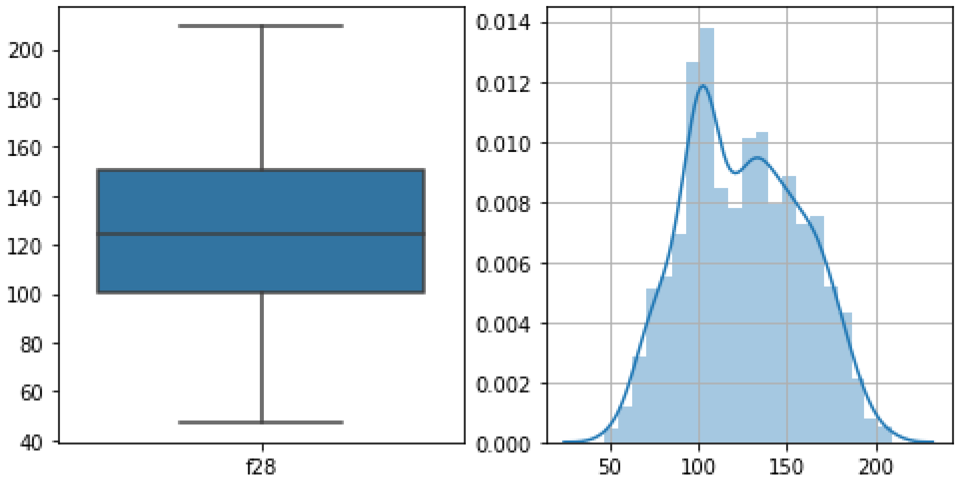


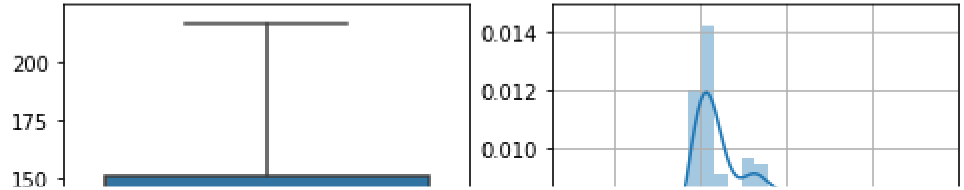
200
180
160
140
120
100
80
60
f24
0.014
0.012
0.010
0.008
0.006
0.004
0.002
0.000
50
100
150
200

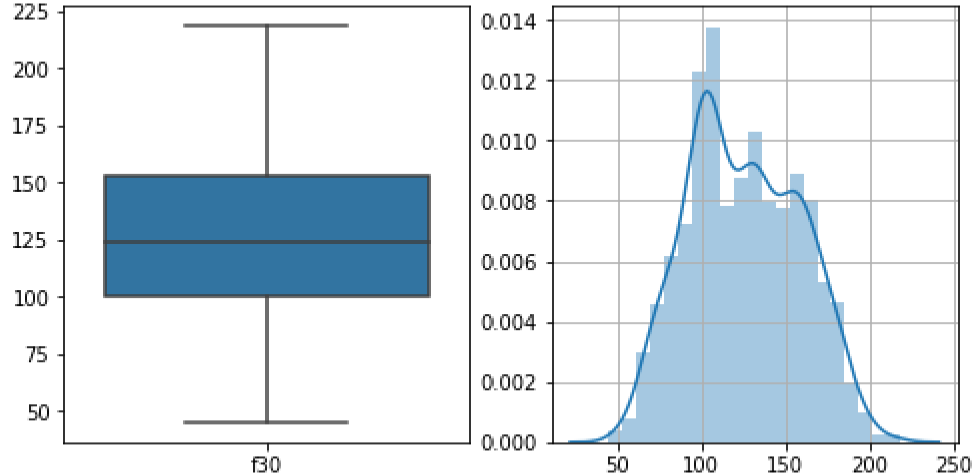

200
175
150
125
100
75
50
f25
0.014
0.012
0.010
0.008
0.006
0.004
0.002
0.000
50
100
150
200

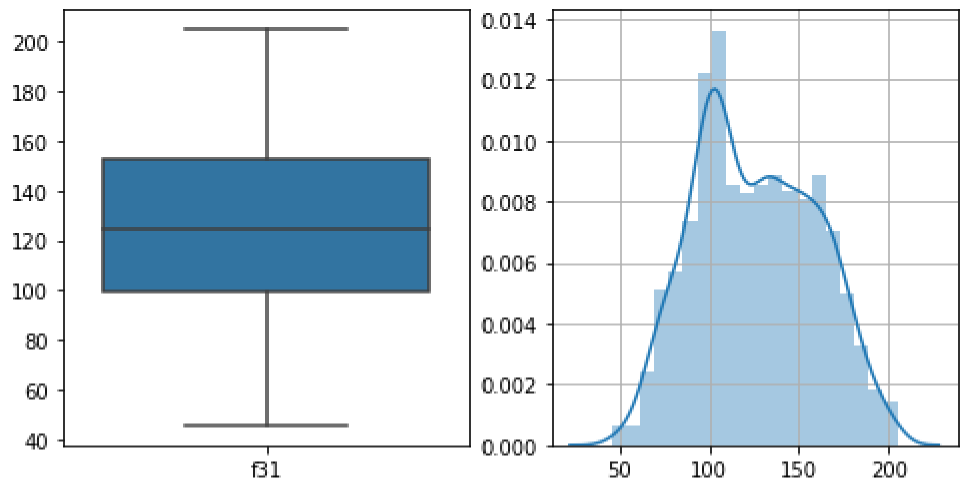

200
175
150
125
100
75
50
f26
0.014
0.012
0.010
0.008
0.006
0.004
0.002
0.000
50
100
150
200

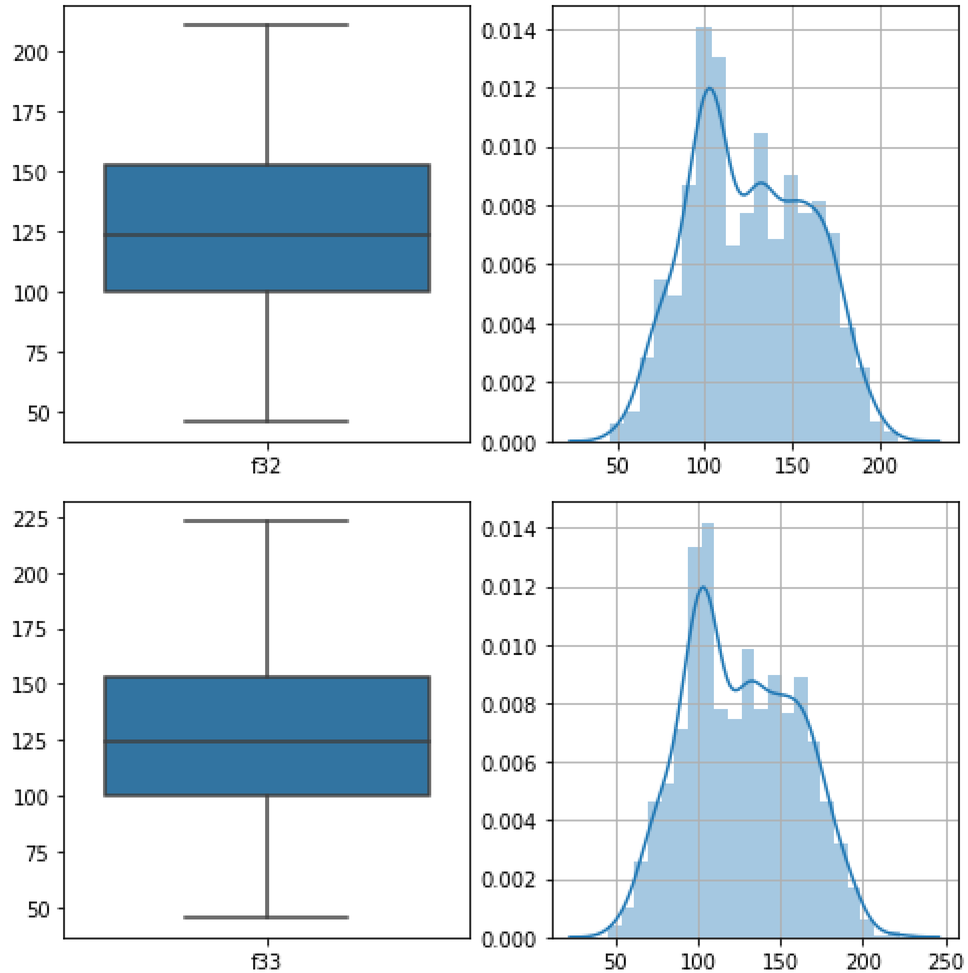

200
180
160
140
120
100
0.014
0.012
0.010
0.008
0.006
0.004


200
180
160
140
120
100
80
60
40
f28
0.014
0.012
0.010
0.008
0.006
0.004
0.002
0.000
50
100
150
200

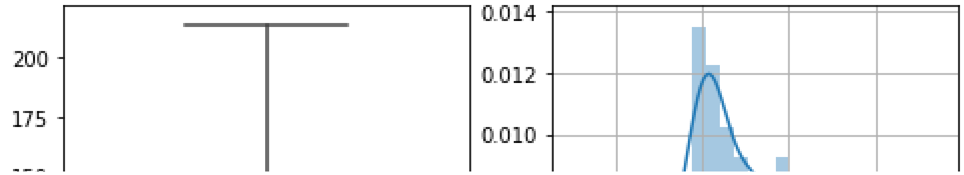

200
175
150
0.014
0.012
0.010

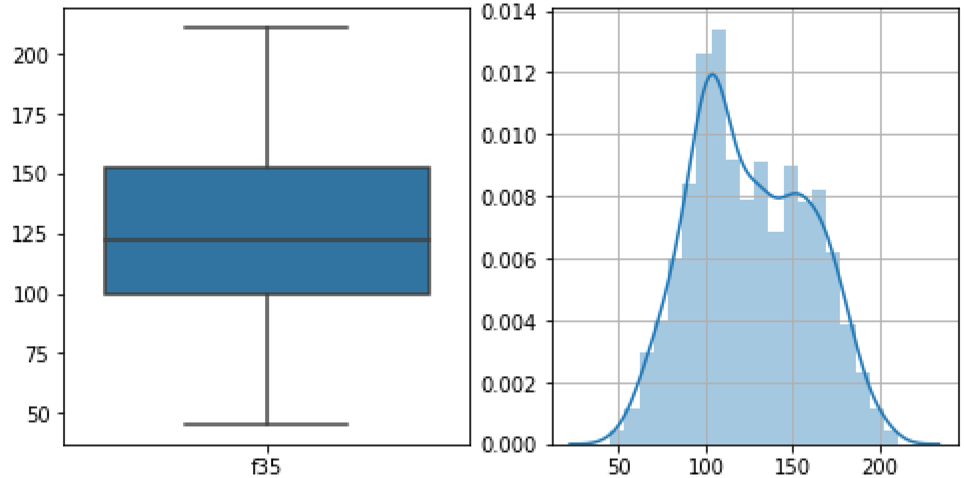

225
200
175
150
125
100
75
50
f30
0.014
0.012
0.010
0.008
0.006
0.004
0.002
0.000
50
100
150
200
250

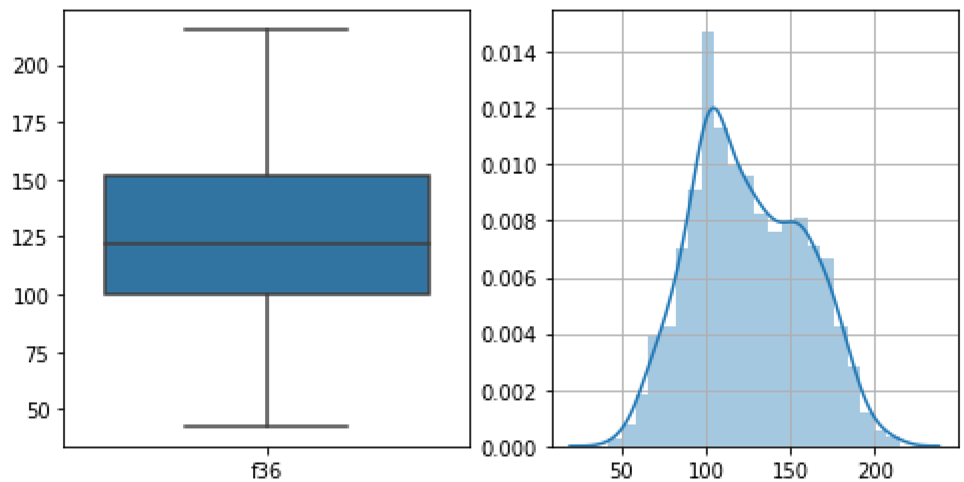

200
180
160
140
120
100
80
60
40
f31
0.014
0.012
0.010
0.008
0.006
0.004
0.002
0.000
50
100
150
200

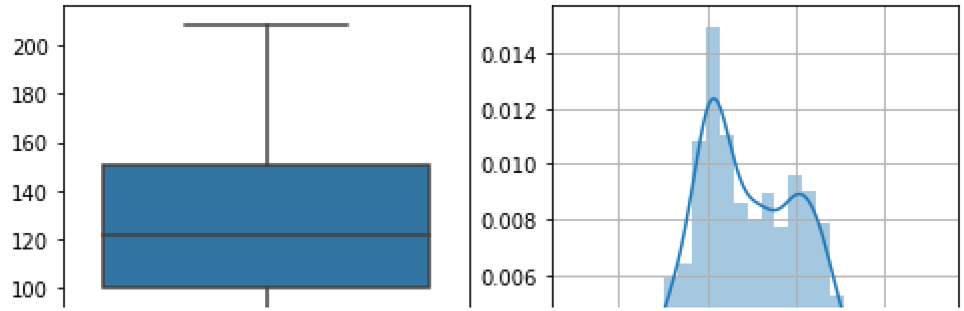


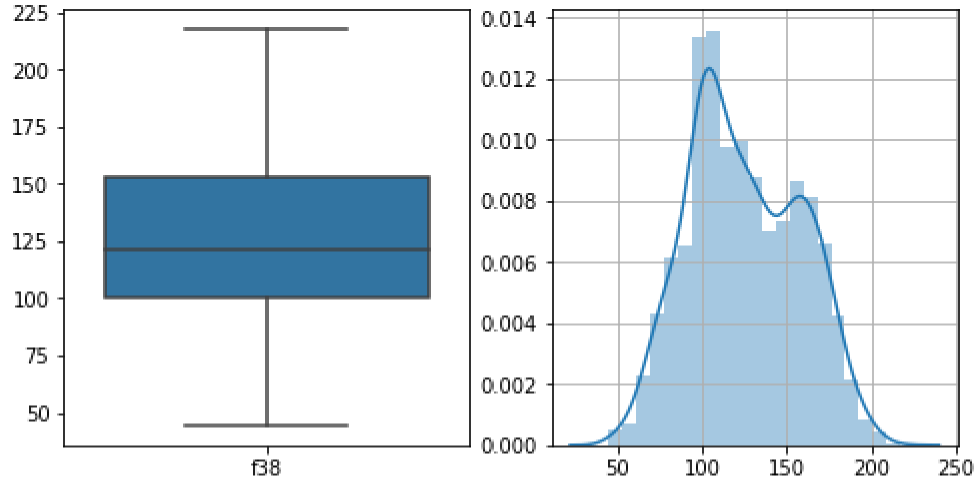
200
175
150
125
100
75
50
f32
0.014
0.012
0.010
0.008
0.006
0.004
0.002
0.000
50
100
150
200
225
200
175
150
125
100
75
50
f33
0.014
0.012
0.010
0.008
0.006
0.004
0.002
0.000
50
100
150
200
250

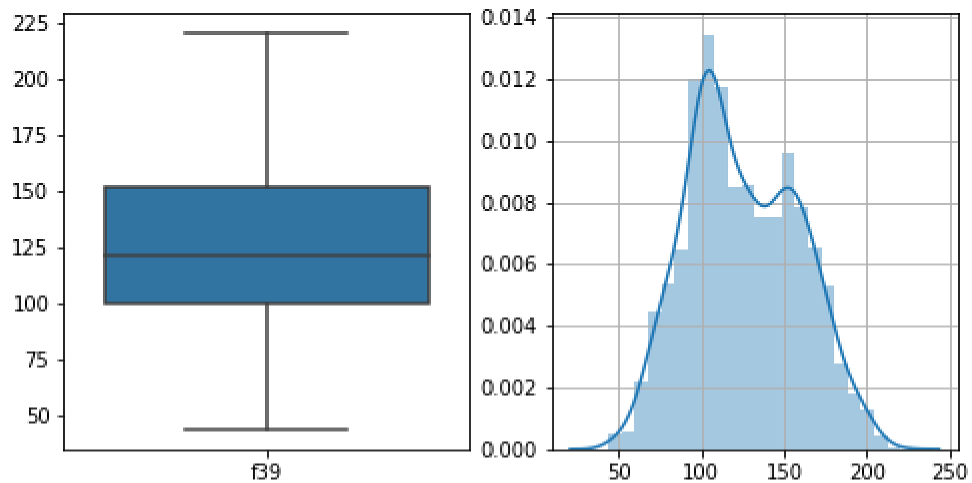

200
175
0.014
0.012
0.010

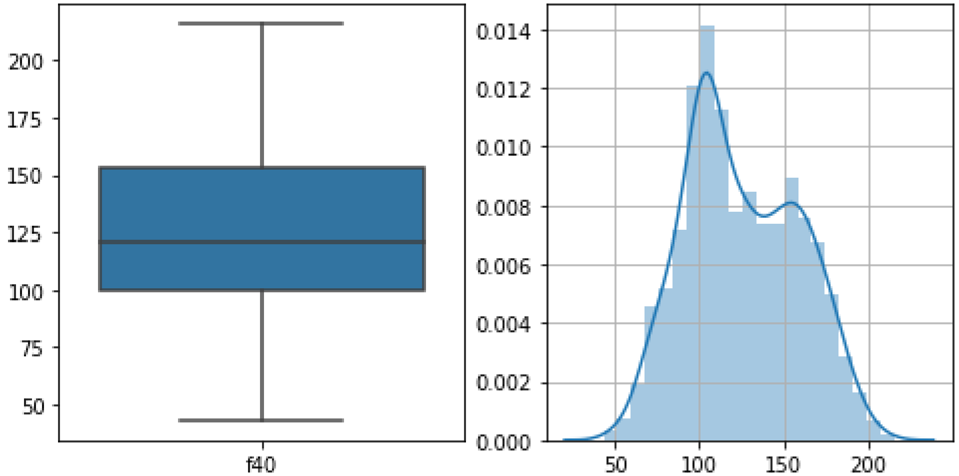

200
175
150
125
100
75
50
f35
0.014
0.012
0.010
0.008
0.006
0.004
0.002
0.000
50
100
150
200

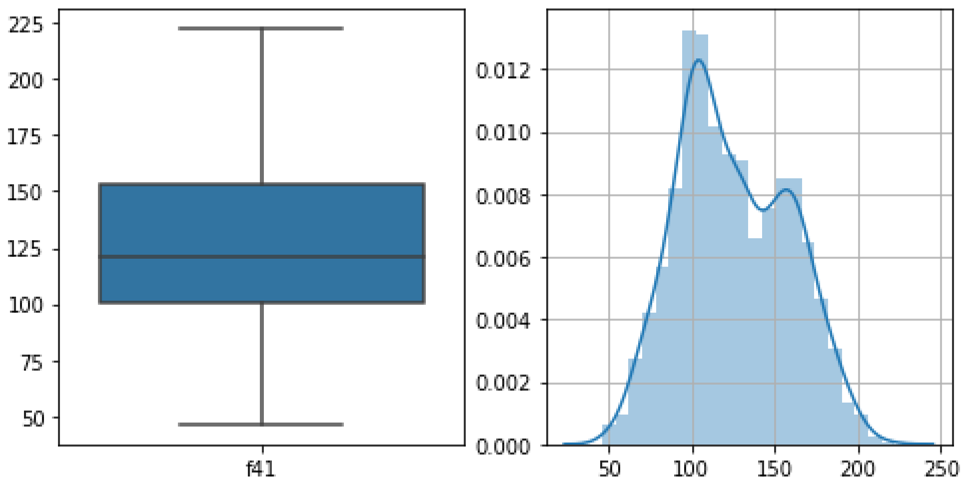

200
175
150
125
100
75
50
f36
0.014
0.012
0.010
0.008
0.006
0.004
0.002
0.000
50
100
150
200

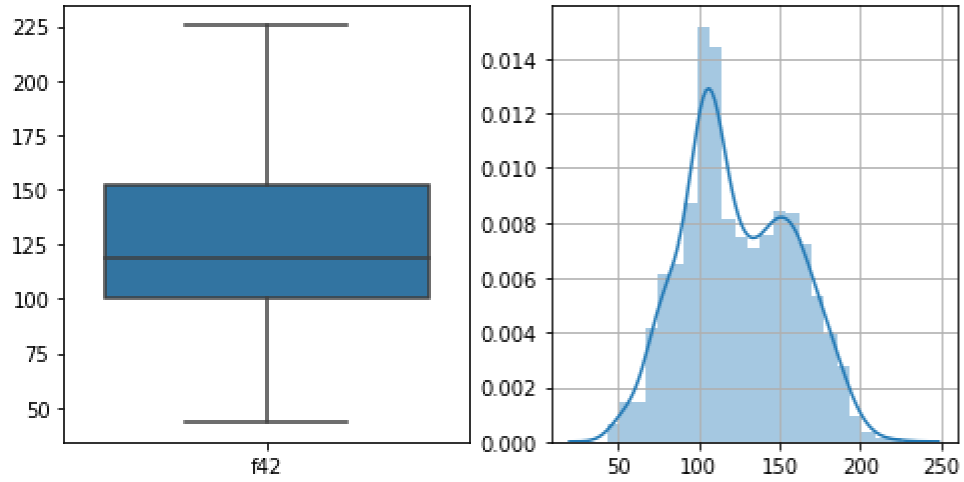

200
180
160
140
120
100
0.014
0.012
0.010
0.008
0.006

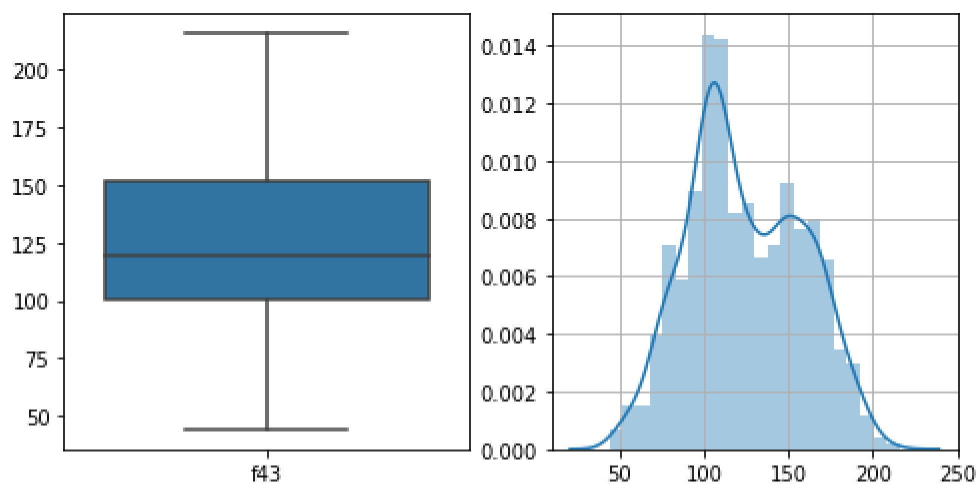

225
200
175
150
125
100
75
50
f38
0.014
0.012
0.010
0.008
0.006
0.004
0.002
0.000
50
100
150
200
250

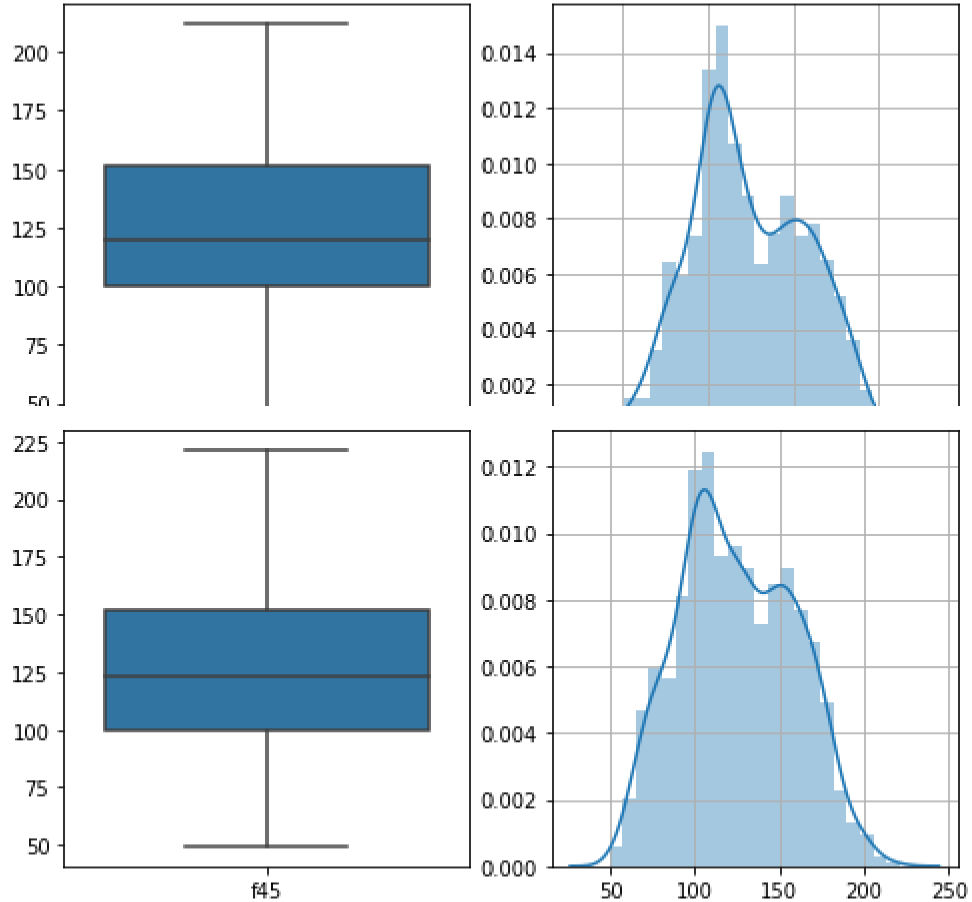

225
200
175
150
125
100
75
50
f39
0.014
0.012
0.010
0.008
0.006
0.004
0.002
0.000
50
100
150
200
250

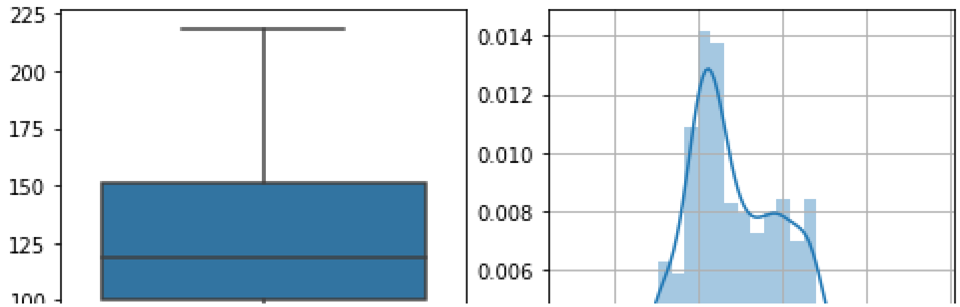

f40

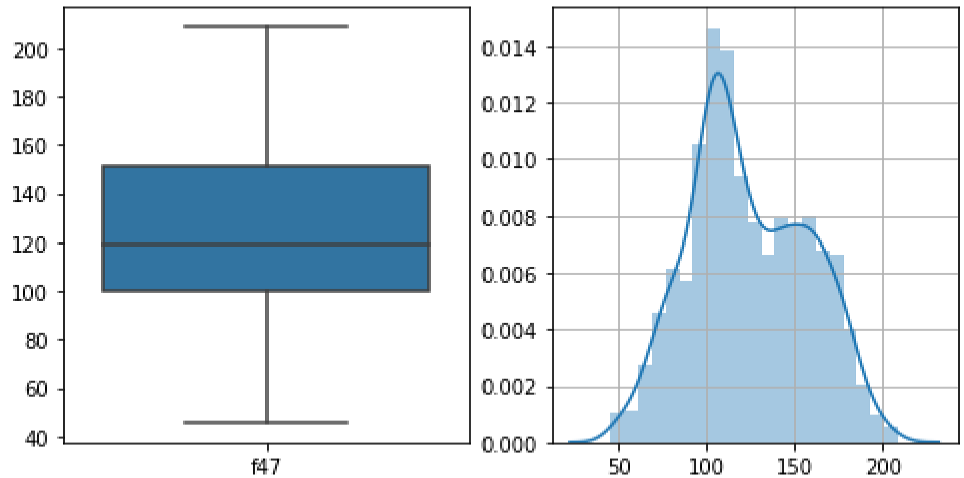

f41

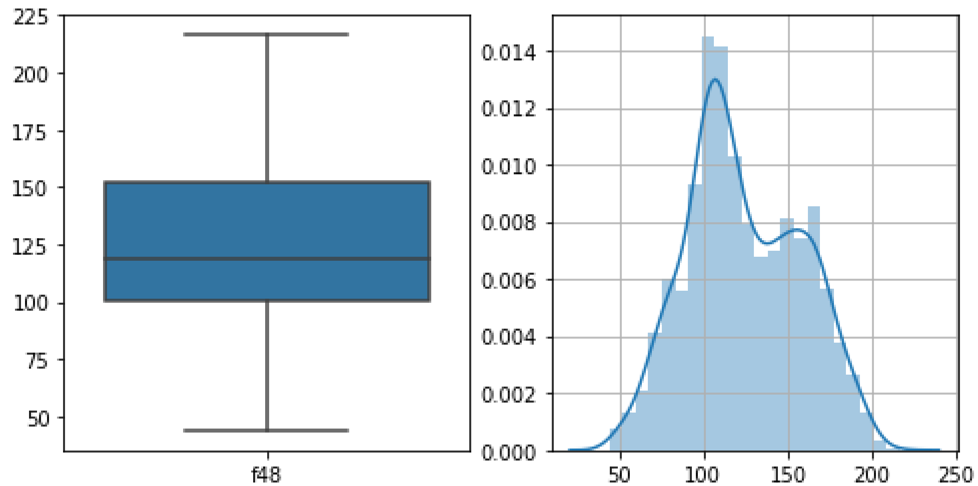

225
200
175
150
125
100
75
50
f42
0.014
0.012
0.010
0.008
0.006
0.004
0.002
0.000
50
100
150
200
250

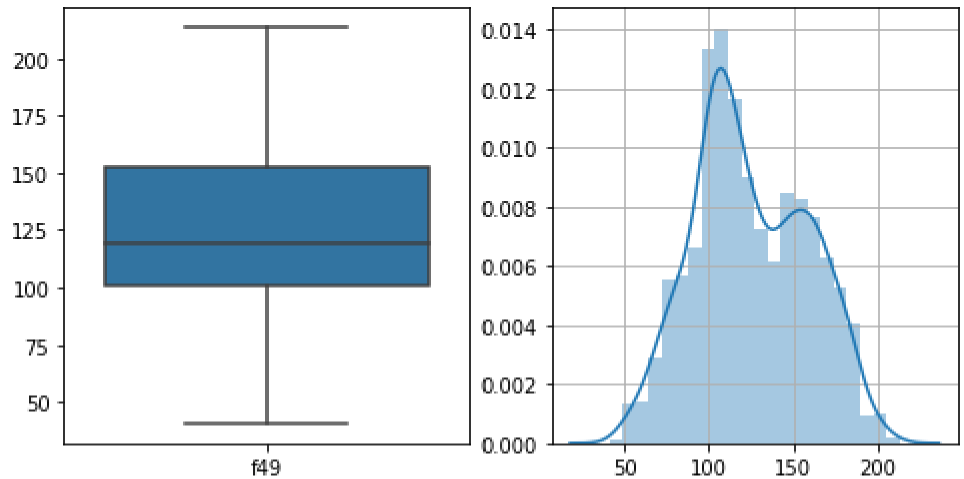

200
175
150
125
100
75
50
f43
0.014
0.012
0.010
0.008
0.006
0.004
0.002
0.000
50
100
150
200
250

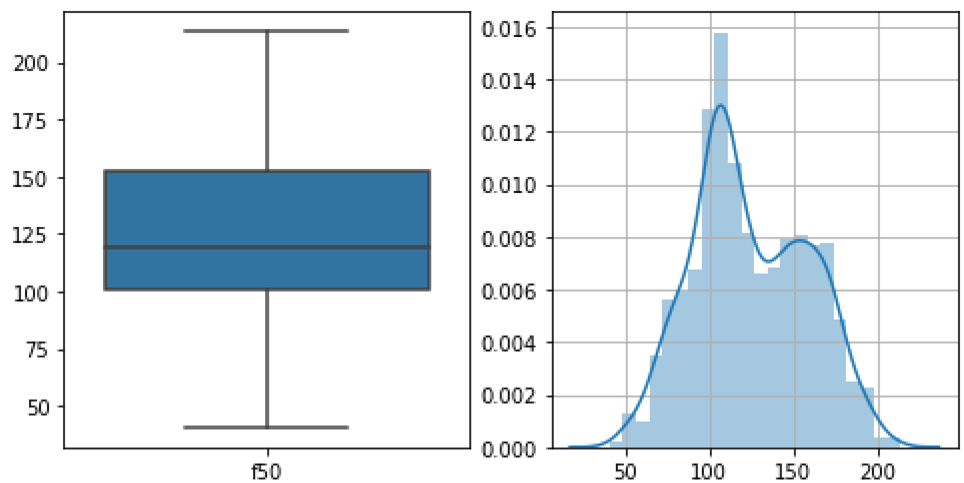

200
175
150
125
100
75
50
0.014
0.012
0.010
0.008
0.006
0.004
0.002
225
200
175
150
125
100
75
50
f45
0.012
0.010
0.008
0.006
0.004
0.002
0.000
50
100
150
200
250

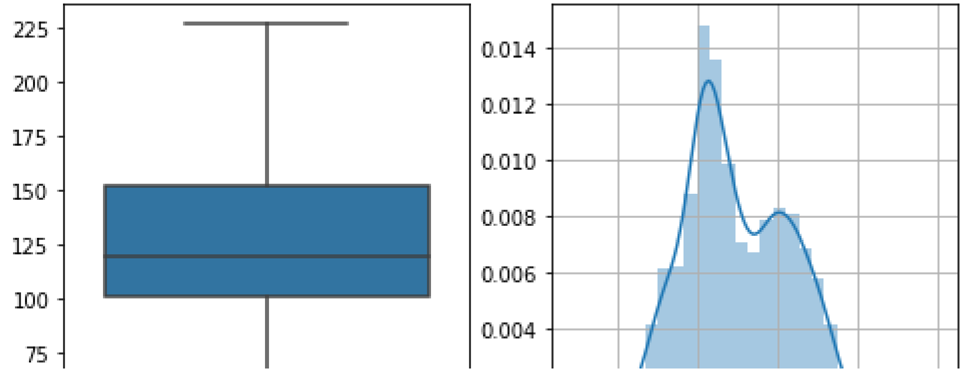

225
200
175
150
125
0.014
0.012
0.010
0.008
0.006

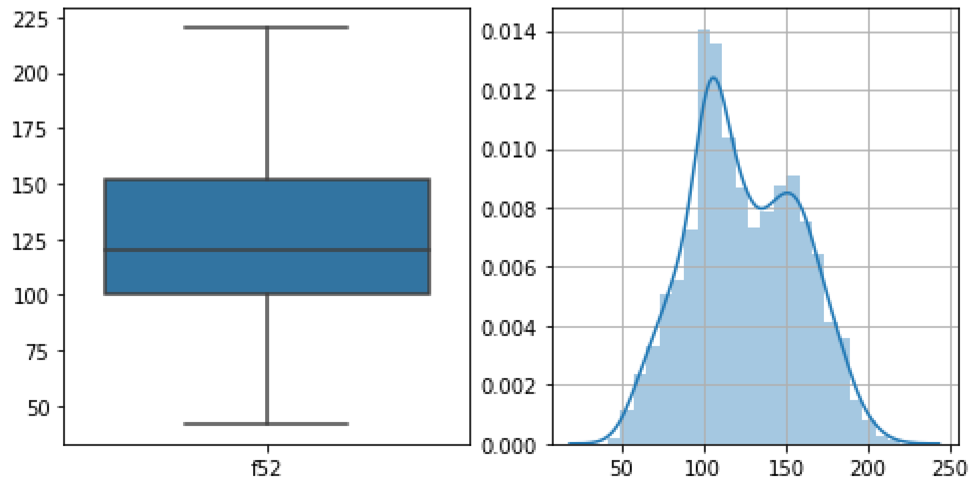

200
180
160
140
120
100
80
60
40
f47
0.014
0.012
0.010
0.008
0.006
0.004
0.002
0.000
50
100
150
200

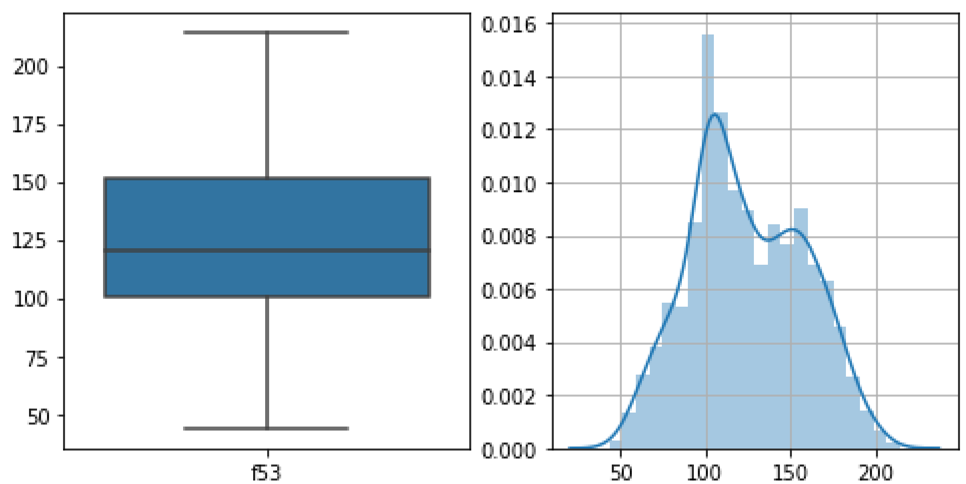

225
200
175
150
125
100
75
50
f48
0.014
0.012
0.010
0.008
0.006
0.004
0.002
0.000
50
100
150
200
250

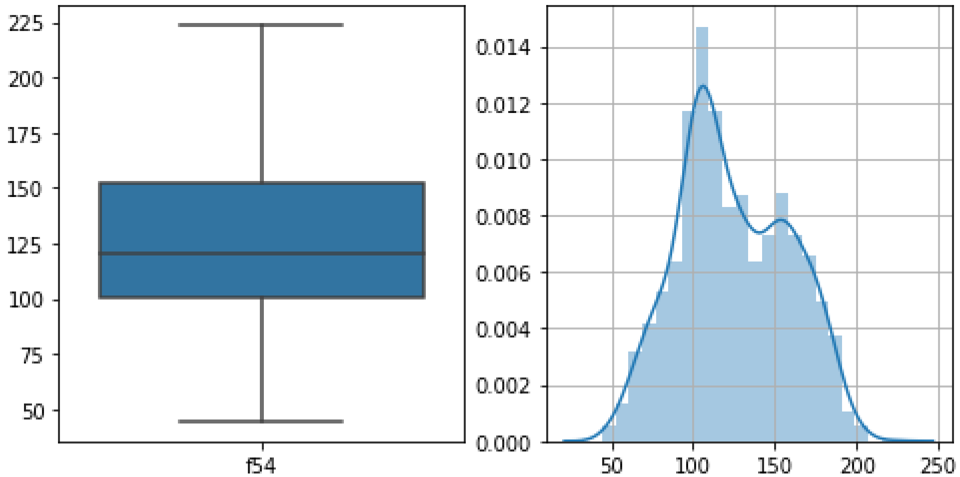

200
175
150
125
100
75
50
f49
0.014
0.012
0.010
0.008
0.006
0.004
0.002
0.000
50
100
150
200

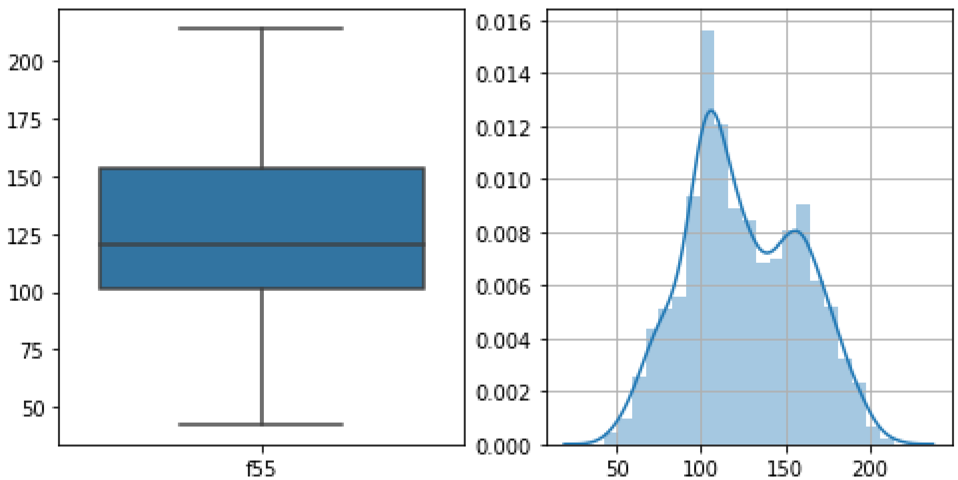

200
175
150
125
100
75
50
f50
0.016
0.014
0.012
0.010
0.008
0.006
0.004
0.002
0.000
50
100
150
200

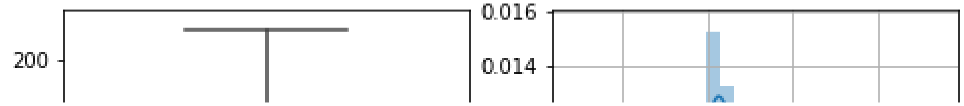

225
200
175
150
125
100
75
0.014
0.012
0.010
0.008
0.006
0.004

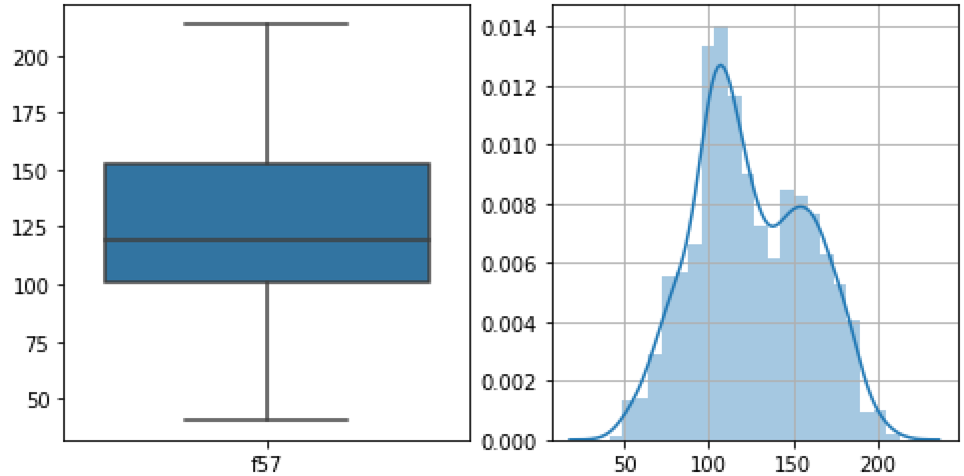

225
200
175
150
125
100
75
50
f52
0.014
0.012
0.010
0.008
0.006
0.004
0.002
0.000
50
100
150
200
250

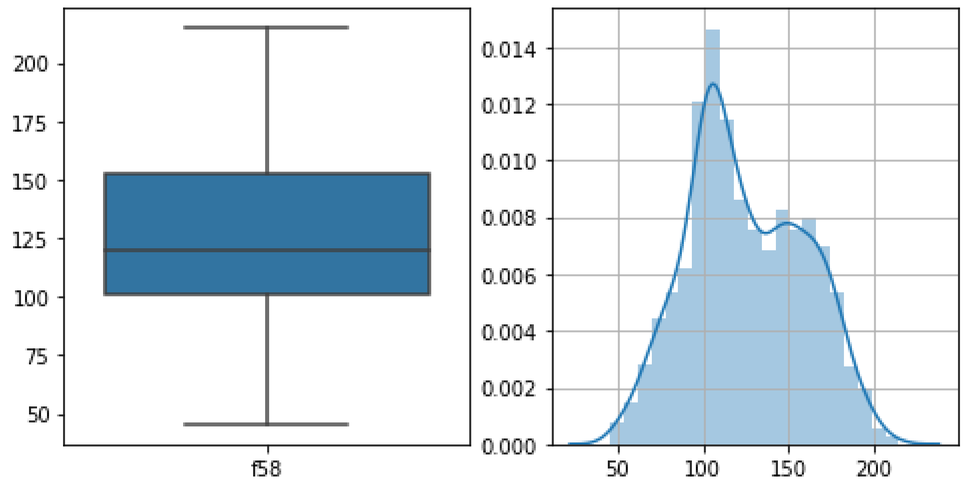

200
175
150
125
100
75
50
f53
0.016
0.014
0.012
0.010
0.008
0.006
0.004
0.002
0.000
50
100
150
200

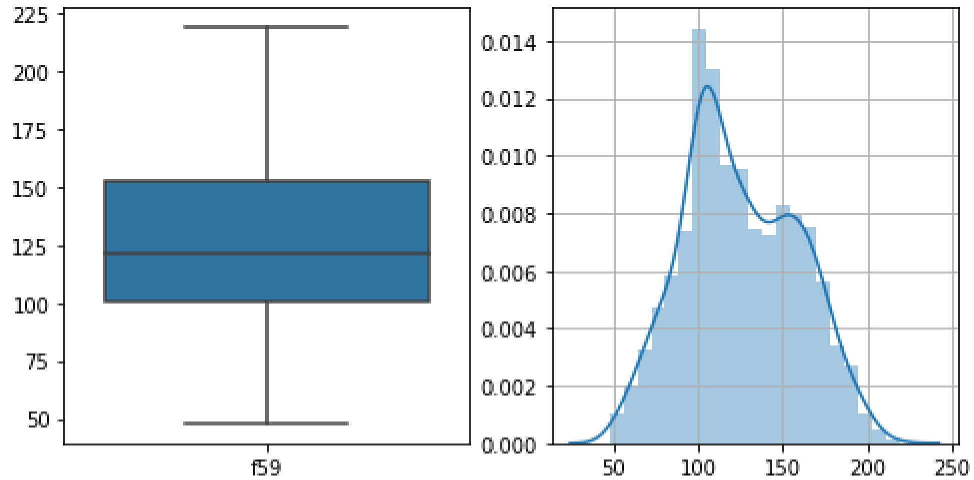

225
200
175
150
125
100
75
50
f54
0.014
0.012
0.010
0.008
0.006
0.004
0.002
0.000
50
100
150
200
250

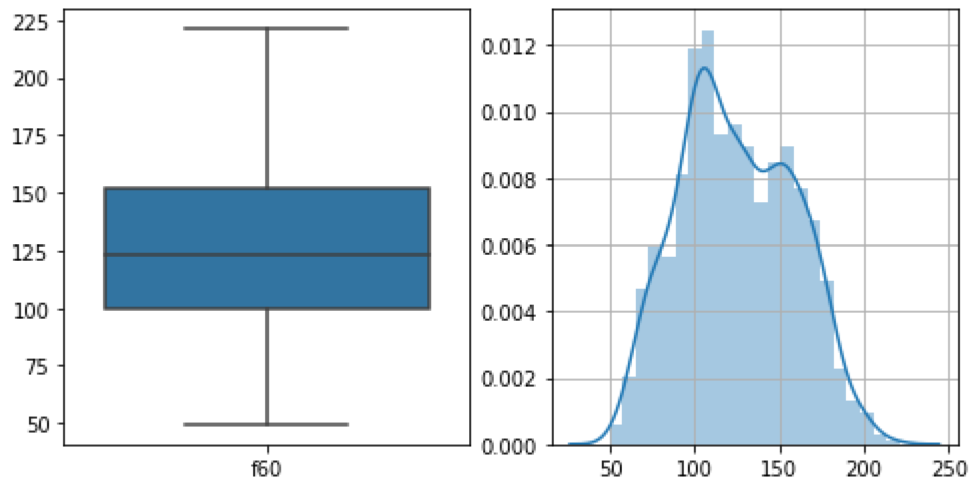

200
175
150
125
100
75
50
f55
0.016
0.014
0.012
0.010
0.008
0.006
0.004
0.002
0.000
50
100
150
200

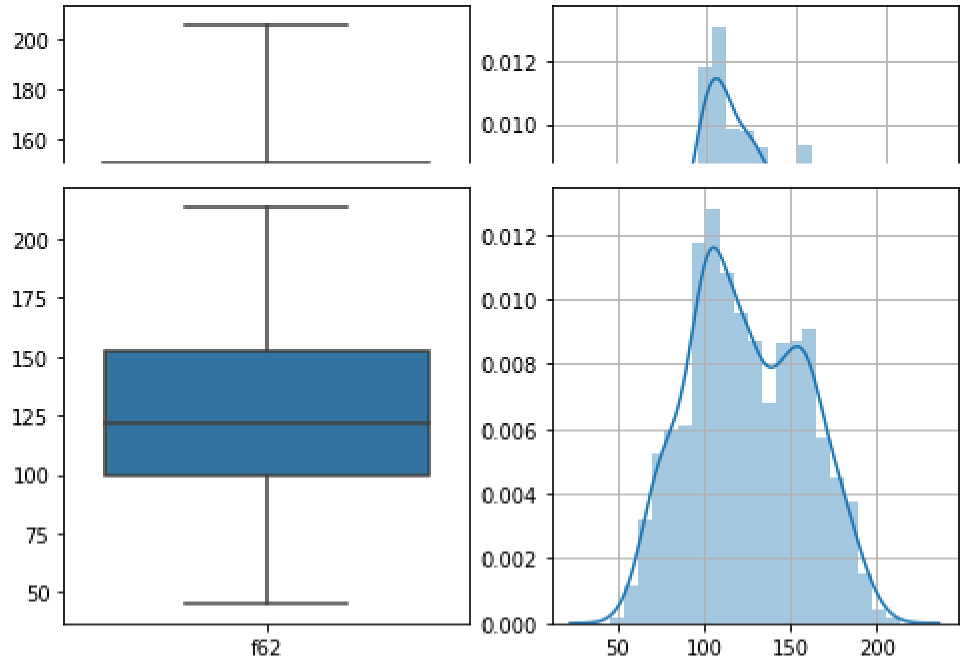

200
0.016
0.014

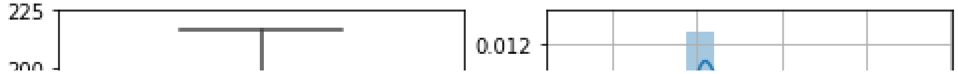

200
175
150
125
100
75
50
f57
0.014
0.012
0.010
0.008
0.006
0.004
0.002
0.000
50
100
150
200

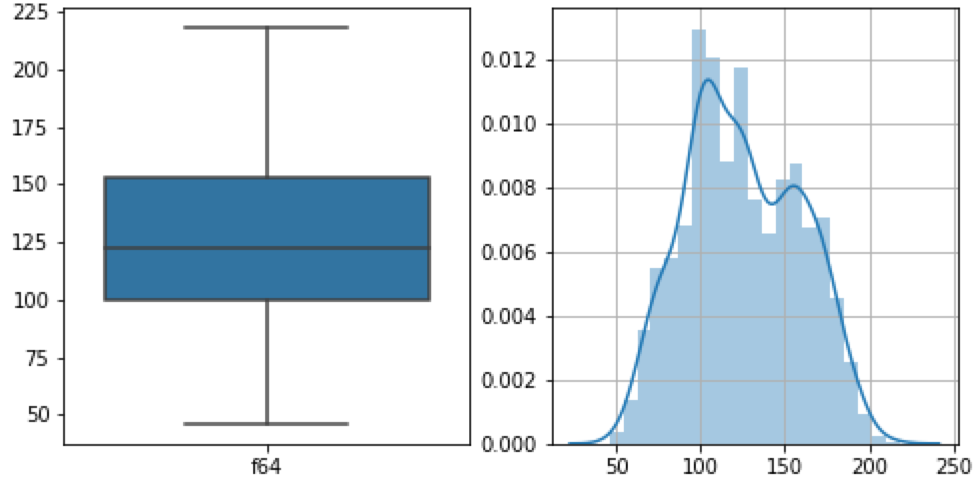

200
175
150
125
100
75
50
f58
0.014
0.012
0.010
0.008
0.006
0.004
0.002
0.000
50
100
150
200

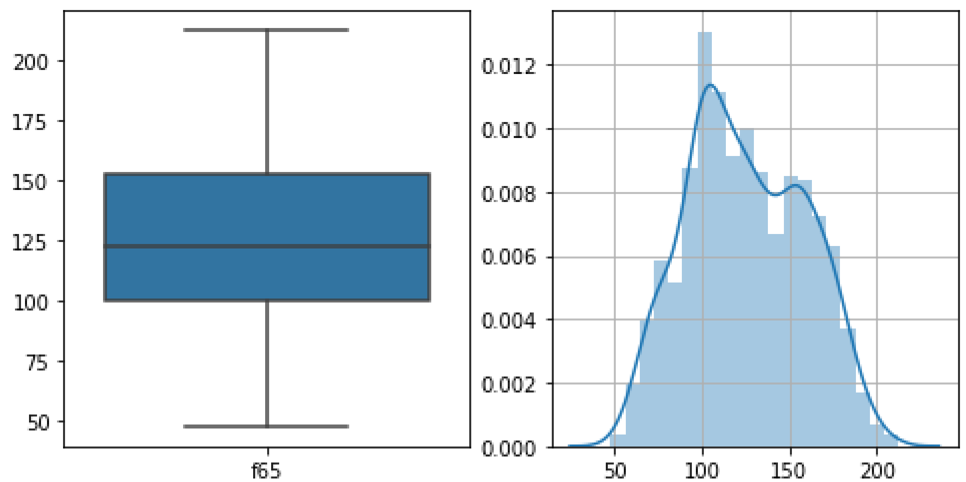

f59
50
75
100
125
150
175
200
225
0.000
0.002
0.004
0.006
0.008
0.010
0.012
0.014
50
100
150
200
250

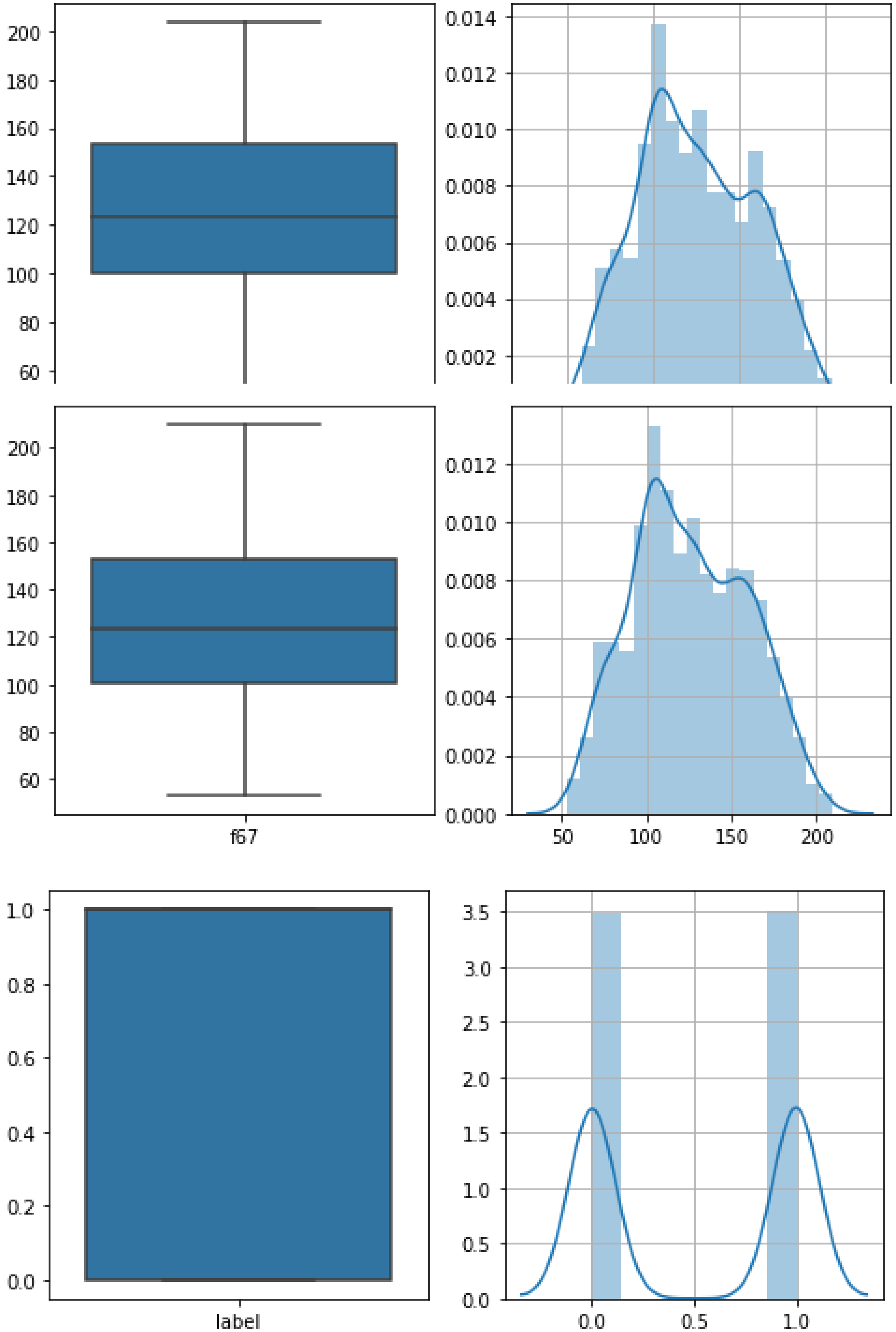

f60
50
75
100
125
150
175
200
225
0.000
0.002
0.004
0.006
0.008
0.010
0.012
50
100
150
200
250


f62




f64


f65


200
180
160
140
120
100
80
60
0.014
0.012
0.010
0.008
0.006
0.004
0.002
f67
0.000
50
100
150
200
1.0
0.8
0.6
0.4
0.2
0.0
label
3.5
3.0
2.5
2.0
1.5
1.0
0.5
0.0
0.5
1.0

Step 5 - Using cw_stats (cw.describe) to look for duplicate columns

In [48]:

```
#convert cw_stats to use in loop to find duplicate columns
n = np.array(cw_stats.select('*').collect())
n[1:2]
```

Out[48]:

```
array([['mean', '125.56923144531228', '125.3568066406251',
        '125.42204101562524', '125.71933251953112', '126.0050551757814',
        '126.08446533203112', '125.88512255859395', '125.7838330078125',
        '125.70476562499994', '125.8583608398437', '125.3948452148437',
        '125.67408691406261', '126.01381738281238', '126.07101806640624',
        '125.90554833984386', '125.64936181640618', '125.72060742187496',
        '124.63611621093749', '125.92546679687484', '125.6720952148438',
        '125.62689404296877', '126.12082763671881', '126.45232910156246',
        '126.03068115234356', '125.9389218750001', '125.43698535156238',
        '125.6427724609375', '125.35043408203126', '125.1534243164061',
        '125.5887226562502', '125.86260253906245', '125.77710107421893',
        '126.11436035156238', '124.8027563476562', '125.48534863281249',
        '125.10084667968752', '124.5295268554687', '124.95900634765609',
        '125.344212890625', '125.35271582031214', '125.32281494140634',
        '124.36889697265605', '124.49470117187522', '124.37970361328104',
        '124.98884716796871', '124.21263525390616', '124.01551074218737',
        '124.23988769531276', '125.0545766601565', '124.72619677734347',
        '124.85175195312473', '124.68179931640644', '124.55499755859377',
        '124.78828515624977', '124.98818603515645', '124.72449658203108',
        '125.0545766601565', '125.26291943359394', '125.20168164062505',
        '124.98884716796871', '124.63611621093749', '124.84174414062501',
        '125.06729882812482', '125.20279980468747', '125.44442382812525',
        '125.41438769531257', '125.56461132812493', '0.50146484375']],
      dtype='<U18')
```

In [49]:

```
#while loop that compares each column with the others for same mean, std, min and max, if same values in st
#initialize empty list for duplicated columns
dup_col = []
#start at 'f1' column
i = 1
while i < 67: #iterate through columns
  j = i +1    #column next to i
  while j < 67:
    if n[1,i]==n[1,j] and n[2,i] == n[2,j] and n[3,i] == n[3,j] and n[4,i] == n[4,j]: #compare values in me
      print(('f') +str(i) +' has same value as' + ' ' + 'f'+str(j))       #if above line is true print form
      col = (('f') + str(i))                                               #if above line is true write
      dup_col.append(col)                                                  #add duplicated col name to a
    j += 1                                                                 #go to next column
  i += 1                                                                   #go to next column
```

```
f18 has same value as f61
f45 has same value as f60
f49 has same value as f57
```

The duplicate columns are f18 and f61, f45 and f60, and f49 and f57.

In [50]:

```
# show dup_col, these columns are duplicates
dup_col
```

Out[50]:

```
['f18', 'f45', 'f49']
```

In [51]:

```
#compare columns visually (just looking at some rows to verify)
cw['f18', 'f61'].show(30)
cw['f45', 'f60'].show(30)
cw['f49', 'f57'].show(30)
```

```
+------+------+
|   f18|   f61|
+------+------+
|161.06|161.06|
| 141.0| 141.0|
|167.11|167.11|
|152.43|152.43|
|167.31|167.31|
|137.61|137.61|
|154.72|154.72|
|147.58|147.58|
|121.19|121.19|
|125.42|125.42|
|132.31|132.31|
|117.99|117.99|
|170.24|170.24|
|159.85|159.85|
| 182.1| 182.1|
|115.72|115.72|
|142.22|142.22|
|153.77|153.77|
|132.35|132.35|
|132.08|132.08|
|170.31|170.31|
|145.57|145.57|
|129.51|129.51|
|141.76|141.76|
|183.05|183.05|
|153.98|153.98|
|128.21|128.21|
|162.81|162.81|
|112.29|112.29|
|106.02|106.02|
+------+------+
only showing top 30 rows

+------+------+
|   f45|   f60|
```

```
+------+------+
|157.51|157.51|
|153.39|153.39|
|155.37|155.37|
|157.83|157.83|
|135.74|135.74|
|142.59|142.59|
| 174.9| 174.9|
|145.76|145.76|
|142.04|142.04|
|121.89|121.89|
|152.21|152.21|
|111.65|111.65|
|166.09|166.09|
|140.55|140.55|
|173.09|173.09|
|116.58|116.58|
|152.08|152.08|
|132.77|132.77|
|144.96|144.96|
|144.15|144.15|
|173.24|173.24|
|144.03|144.03|
|121.02|121.02|
|136.09|136.09|
|165.04|165.04|
|159.47|159.47|
|157.56|157.56|
|154.11|154.11|
|110.73|110.73|
|92.421|92.421|
+------+------+
only showing top 30 rows

+------+------+
|   f49|   f57|
+------+------+
|162.24|162.24|
|155.39|155.39|
|141.61|141.61|
|138.08|138.08|
|149.46|149.46|
|153.07|153.07|
```

```
|144.44|144.44|
|155.85|155.85|
|142.59|142.59|
|158.22|158.22|
|144.47|144.47|
|148.74|148.74|
|166.72|166.72|
|146.49|146.49|
|147.42|147.42|
|152.76|152.76|
|155.28|155.28|
|152.68|152.68|
|115.43|115.43|
|134.41|134.41|
|137.97|137.97|
|123.98|123.98|
|145.66|145.66|
|128.35|128.35|
|178.49|178.49|
|149.27|149.27|
|131.51|131.51|
|144.92|144.92|
|108.16|108.16|
| 128.5| 128.5|
+------+------+
only showing top 30 rows
```

In [0]:

```
from pyspark.mllib.stat import Statistics
```

In [53]:

```
#another way to confirm duplicate columns
cw_stats.stat.crosstab('f18','f61').show()
cw_stats.stat.crosstab('f45','f60').show()
cw_stats.stat.crosstab('f49','f57').show()
```

```
+------------------+-----------------+----+-----+-----------------+------+
|           f18_f61|124.63611621093749|2048|205.7|33.20314825083078|47.444|
+------------------+-----------------+----+-----+-----------------+------+
|             205.7|                0|   0|    1|                0|     0|
| 33.20314825083078|                0|   0|    0|                1|     0|
|124.63611621093749|                1|   0|    0|                0|     0|
|              2048|                0|   1|    0|                0|     0|
|            47.444|                0|   0|    0|                0|     1|
+------------------+-----------------+----+-----+-----------------+------+

+------------------+-----------------+----+------+-----------------+------+
|           f45_f60|124.98884716796871|2048|221.07|33.52747095890803|49.456|
+------------------+-----------------+----+------+-----------------+------+
|            221.07|                0|   0|     1|                0|     0|
|            49.456|                0|   0|     0|                0|     1|
|124.98884716796871|                1|   0|     0|                0|     0|
|              2048|                0|   1|     0|                0|     0|
| 33.52747095890803|                0|   0|     0|                1|     0|
+------------------+-----------------+----+------+-----------------+------+

+------------------+----------------+----+-----+-----------------+------+
|           f49_f57|125.0545766601565|2048|213.5|33.813020348244834|40.721|
+------------------+----------------+----+-----+-----------------+------+
|            40.721|               0|   0|    0|                0|     1|
|33.813020348244834|               0|   0|    0|                1|     0|
| 125.0545766601565|               1|   0|    0|                0|     0|
|             213.5|               0|   0|    1|                0|     0|
|              2048|               0|   1|    0|                0|     0|
+------------------+----------------+----+-----+-----------------+------+
```

Because these three columns are duplicates (and I know that they have been added since we did the homework on the same sets) I will drop them because otherwise they would negatively impact the accuracy of any results that would be achieved with later tests. Without the prior knowledge of what a dataset might look like, however, in other scenarios, it would be more difficult to make the decision to drop these columns without further verifying why these three duplicate columns exist.

In [0]:

```python
#using dup_col to drop columns
for x in dup_col:
  cw = cw.drop(x)
```

In [55]:

```python
#show cw with dropped columns
cw.show()
```

```
-----+------+------+------+------+------+------+------+------+------+------+------+------+------+----
   f1|    f2|    f3|    f4|    f5|    f6|    f7|    f8|    f9|   f10|   f11|   f12|   f13|   f14|   f
-----+------+------+------+------+------+------+------+------+------+------+------+------+------+----
70.39|167.28|143.44|124.67|139.01|125.83|144.33|151.26|175.51|171.31| 161.9|146.92| 141.8|140.91| 132
69.75|190.96|175.53|138.27|137.47|139.23|133.23|130.25|147.73|163.93|167.36|171.52|155.54|139.34|151.
53.69|153.68|144.02|158.73|178.87|157.04|152.92|147.52|142.87|165.26|160.39|137.86|149.62|153.43| 152
31.69|151.56|151.05| 134.0|151.18|175.53|171.34|159.77|151.95| 146.1|148.53|140.28|138.16|145.44| 150
62.85|158.88|132.27|138.41|143.98| 159.3|177.26|180.58|159.34|164.66|138.04|132.76|157.88|165.58|173.
32.05|149.12|165.08|170.62|162.19| 157.1|145.86|149.52|162.84| 149.5|138.86|140.41|156.82|171.41|158.
53.59|142.25|157.33|156.08|149.33|162.97|150.25|146.47|145.99|137.82| 152.9|161.64|150.23| 170.7|185.
67.68|153.49|149.19|148.71|166.03|167.04|153.06|157.48|133.57|143.66|167.27|172.45| 179.8|169.15| 150
36.48|130.02|131.72|152.04|163.03|172.93|170.11| 165.2|166.41|120.67|119.02|135.76|147.52|164.59|176.
45.96|140.31|126.34|113.12|118.66|140.33| 139.9|139.51| 168.7|149.54|149.38|147.88|129.45|143.81|131.
31.08| 123.3|147.32|150.99|150.73|148.27|153.97|157.52|150.63| 128.4|130.98| 150.6|151.69|146.12|149.
170.6|147.29|151.48|149.29| 137.6|162.15|171.54|131.15|165.84|141.37|143.72|138.37| 129.1|160.41|176.
39.11|131.09|135.86|138.26|139.32|129.36|127.15|128.16|140.02|151.06|168.77|156.14|128.71|110.57|117.
53.41|173.65|168.81|143.56|130.35|128.61|138.26|132.69| 142.4|157.31|147.23|164.19|175.49| 153.2|153.
18.62|141.01|152.81|168.03|141.41|124.15|138.97|136.46|111.73|147.31| 160.7|160.62| 153.6|132.28| 135
43.01| 150.6|160.42|160.71| 164.4|152.44|153.82|142.38|121.96|134.68|145.74|164.03|181.31| 169.5|157.
22.28|121.07|136.34|164.09|171.49| 167.5| 153.3|135.84|152.38|127.52|118.93|129.54| 143.0|158.57|151.
144.6|158.24|156.54|130.02| 144.2|146.98|144.31|137.42|151.11|164.84|160.35|147.28|157.32| 169.6|171.
05.11|104.67|117.24|132.64|120.32|119.85|132.25|133.11|124.45|128.66|125.11|150.99|144.23|136.57|146.
44.26| 132.2|140.19|161.19|189.46|157.34|103.98|120.33| 118.7|128.71|144.17|149.85|151.25|149.96| 124
-----+------+------+------+------+------+------+------+------+------+------+------+------+------+----
ly showing top 20 rows
```

In [56]:

```python
#check cw shape, there are now 64 features plus a column for label
print(spark_shape(cw))
```

```
(2048, 65)
```

Step 6 - Using pyspark sql to count label values (is the dataset balanced, complete and

Step 6 - Using pyspark sql to count label values (is the dataset balanced, complete and accurate?)

In [57]:
```
#register cw_stats as cw_stats table
from pyspark.sql import SQLContext
from pyspark import SparkContext as sc
sc = pyspark.SparkContext.getOrCreate()
sqlContext = SQLContext(sc)
#update cw_stats after removing columns in cw
cw_stats = cw.describe()
cw_stats.registerTempTable('cw_stats_table')

sqlContext.sql('select distinct(*) from cw_stats_table').show()
```

```
+-------+------------------+------------------+------------------+------------------+----------------
|summary|                f1|                f2|                f3|                f4|               f
+-------+------------------+------------------+------------------+------------------+----------------
|  count|              2048|              2048|              2048|              2048|             204
| stddev| 33.29273147272022|32.822212055580586| 32.64300489797472| 32.96603101800241|33.5262474593387
|    max|            210.65|             210.2|            212.93|             211.0|            213.
|    min|            47.124|            47.262|            48.485|            49.323|            47.07
|   mean|125.56923144531228| 125.3568066406251|125.42204101562524|125.71933251953112|126.005055175781
+-------+------------------+------------------+------------------+------------------+----------------
```

In [0]:
```
#register cw as cw_table
cw.registerTempTable('cw_table')
#display statistics for rows with value label as '1'
only_ones = sqlContext.sql('select * from cw_table where label = 1')
```

In [59]:
```
only_ones.describe().show()
```

```
+-------+------------------+-----------------+------------------+------------------+------------------+-
|summary|                f1|               f2|                f3|                f4|                f5|
+-------+------------------+-----------------+------------------+------------------+------------------+-
|  count|              1027|             1027|              1027|              1027|              1027|
|   mean|  99.4509055501461|99.59365043816952| 99.68817526777008| 99.61038656280431| 99.54279065238548|
| stddev|18.165017548622206|18.23404010576009|18.234327451342406|18.247597194850368|18.284510087299147|1
|    min|            47.124|           47.262|            48.485|            49.323|            47.077|
|    max|            175.99|           162.87|            186.07|            170.71|            158.59|
+-------+------------------+-----------------+------------------+------------------+------------------+-
```

In [60]:
```
#use cw_table to count rows with label of '0'
sqlContext.sql('select count(*) as Label_is_0 from cw_table where label = 0').show()
```

```
+----------+
|Label_is_0|
+----------+
|      1021|
+----------+
```

In [61]:
```
#use cw_table to show label count, there are more label values of '1' than label values of '0'
sqlContext.sql('select label, count(*) as Label from cw_table group by label').show()
```

```
+-----+-----+
|label|Label|
+-----+-----+
|    1| 1027|
|    0| 1021|
+-----+-----+
```

To discuss question of imbalanced, inaccurate and or incomplete data, checking if there are duplicate rows by creating a duplicate row table using group by and converting to table for sql querying

In [0]:

```
#create an additional column named "e" and assign 0 if count of row is 1 and 1 if the count of row is great
dup_rows = cw.groupBy(cw.columns).agg((count("*")>1).cast("int").alias("e"))
```

In [0]:

```
#converting dup_rows to dup_rows_table to query with sql
dup_rows.registerTempTable('dup_rows_table')
```

In [64]:

```
#if dup_rows had duplicate rows, then e would equal one. 2048 rows have '0' as value for e, therefore there
sqlContext.sql('select e, count(*) as Label from dup_rows_table group by e').show()
```

```
+---+-----+
|  e|Label|
+---+-----+
|  0| 2048|
+---+-----+
```

In [65]:

```
dup_rows.show()
```

```
+------+------+------+------+------+------+------+------+------+------+------+------+------+------+-----
|    f1|    f2|    f3|    f4|    f5|    f6|    f7|    f8|    f9|   f10|   f11|   f12|   f13|   f14|   f1
+------+------+------+------+------+------+------+------+------+------+------+------+------+------+-----
|165.19|179.07|184.55|146.75|157.34| 177.4|178.61|154.41|160.62|132.04|133.04|147.13|172.41|179.96|183.6
|156.01|157.13|154.59|141.55|121.44|128.87|164.33|177.31|150.76| 145.8|159.46|175.38|165.23|132.29|125.4
|123.96|114.66| 105.7|110.94|143.88|162.92|174.81| 159.2|126.73|121.65|127.12|106.55|96.744|146.04|174.6
|129.09|125.91|120.81|117.33|132.84|134.89|123.69|133.57|152.66|149.72|133.85|121.71|136.29|122.66|100.7
|116.74|120.05|138.63|127.22| 122.4|149.62|159.81|125.69|149.37|155.68|168.23|166.44|135.56|120.85|109.4
|117.35|118.53|117.28|116.15| 117.6| 119.4|118.38|121.42|114.95|120.48|116.59|116.94|116.98|113.52|116.4
|114.24|115.98|115.77|114.35|116.56|115.55|115.08|118.93|125.09| 125.4|124.94|124.24|125.25| 125.1|124.9
|78.041|79.273|81.262|78.516|76.519|76.987|74.697|77.203|75.455|76.381|76.042|77.304|78.649|76.556|76.66
|110.05|110.42|109.52|111.22|106.53|103.95|102.49|97.689|113.21|112.51|112.33|108.49| 107.3|102.67|99.83
|156.13|179.02|192.41|167.69| 150.8|164.61|158.09|147.75|149.05|175.97|194.11|181.87|161.82|178.59|165.6
| 119.7|157.58|169.68|173.42|155.26|122.91|131.31|169.68|146.75|175.27|186.48|188.29|173.56|139.31|128.6
|137.15| 124.3|149.38| 177.4| 175.3|135.77| 136.6|168.62|105.07|117.86|148.72|162.27|149.12|122.03|138.5
|129.49|119.99|131.91|143.94|144.42|131.97|118.75|142.24|146.09|149.39|152.54|151.31|135.58|153.16|160.2
|131.02|126.95|142.31|165.23|176.81|156.03|155.43|165.37|112.52|127.85|156.84| 158.1|165.71|132.01|122.5
|140.06|160.04|186.07|170.71|158.59|149.36|120.46|107.76|137.16| 160.7|188.36|182.88|154.25|134.01|127.9
|89.331|92.005|94.837|94.161|98.092|95.873|94.497| 93.78| 93.51|94.695|91.506|92.681|93.129|92.933|91.02
| 131.9|131.25|127.78|124.46|125.29|128.75|130.42| 130.6|132.69|132.06|130.92|129.76|127.84|129.67|130.6
|79.557|78.101|75.796|76.927|78.446|81.763|83.777|83.575|82.381|80.758|79.185|80.575|79.339|80.328|82.42
|93.321|89.277|88.065|87.596| 81.19|81.686|87.715|89.988|91.898|90.806|88.211|88.348|86.712|86.724|90.44
| 175.8|184.09| 164.2|129.62|129.46|154.96|160.58| 153.5|169.82|187.82|175.31|138.72| 131.4|157.61|145.2
+------+------+------+------+------+------+------+------+------+------+------+------+------+------+-----
only showing top 20 rows
```

Step 6 continued: Is the data complete, balanced and accurate?

IMBALANCED: The previous cells identified that there were more labels with the value of 1 than the value of 0. There were 1027 cases of '1' in the label category and 1021 cases of '0'. In the case of this dataset, since we constructed it during the last assignment, the expectation would be that there would be an equal number of 0's and 1's (1024 each.) Hence, it is likely that one of two things has happened to the dataset: either entire rows have been deleted and then others duplicated or some 0's have been changed to 1.

INACCURATE: In this case, I would inference that there are three rows that have been altered since the cells used to check for duplicate rows did not indicate any duplicated rows. Unfortunately,I don't think there would be a way to detect which ones had been altered. Thus, in this case, I would find that the dataset is both inaccurate and inbalanced. Additionally, the presence of three duplicated columns also indicates that the data is inaccurate (since I know that there should be 64 columns plus a label and that three columns have been copied.)

COMPLETE: If those rows were completely deleted and others added, then I would find that the data was also incomplete since information would be missing. However, given the above discussion, and since there are no NaN, null or missing values observed the data is complete (in terms of missing items) but it is imbalanced and inaccurate.

Step 7 - reshuffling dataset, dividing dataset into 75:25 train and test sets, save to csv

In [67]:

```
cw.orderBy(rand()).show(20)
```

```
------+------+------+------+------+------+------+------+------+------+------+------+------+------+-----
   f51|   f52|   f53|   f54|   f55|   f56|   f57|   f58|   f59|   f60|   f61|   f62|   f63|   f64|   f6
------+------+------+------+------+------+------+------+------+------+------+------+------+------+-----
 158.4|158.56|156.31|146.56|127.31|136.18|151.44|152.82|164.85|150.65|150.44|156.85|129.97|113.73|136.9
 164.5|150.85|164.16|174.85| 154.8| 138.6| 158.1| 144.7|142.33|164.87|146.21|117.67|102.45|119.43| 133.
92.522|89.176|92.262|91.666|90.146|89.623|86.742| 88.82|93.671|92.162|91.778|95.059|92.023|90.437|94.08
136.51|89.922|93.745|132.81|149.84|135.08| 108.2|99.633|116.97|99.781| 79.94|107.69|124.04|130.11|114.6
145.34|181.37|183.48|168.16|136.09|157.05|176.92|178.45|157.81| 179.6|179.19|178.63|153.51|152.91|173.0
 115.4|107.89|107.95|108.03|108.95|112.58|115.18|112.19|112.02|92.785|95.026|98.964|101.97|103.84| 106.
172.55|201.05|190.06|155.83|136.63|136.17|154.98|143.55|145.42|180.25|157.58|165.33|179.83|173.21| 177.
101.57|103.29|100.63|99.834|101.11| 95.83|94.115|96.402|95.981|98.992|98.679|99.665|101.66|102.58|97.03
156.35|119.05|105.68|97.423|109.41|140.12|166.18|150.03|152.77|141.47|116.23|102.95|117.58|131.18|152.6
 119.5|117.88|118.46|114.66|113.47|116.45|116.27|117.14| 116.0|122.53|117.05|118.59|121.11|120.87|119.9
171.83|181.54|159.51|163.81|178.45|171.82|167.43|171.25|145.79|192.04|149.36|152.06|175.54| 194.6|179.3
103.25|103.34|102.59|102.65|101.38|98.502|96.831|99.385| 99.99|106.88|103.29|103.01| 102.9|102.66|98.94
100.39|100.85|100.08|104.13|103.84|101.28|99.873|99.305|101.38|101.09|103.06|103.02|107.57| 104.2|100.1
64.668|122.61|120.22|127.49|137.85|155.29|168.61|141.46|104.11|116.17|123.97|124.63|124.66| 127.8|137.7
111.29|111.21|108.08|105.35|105.93| 105.7|104.91|106.55|108.95|106.82| 106.4|104.05|103.38|102.34|103.7
158.26|115.46|125.06|144.35|146.31|120.96| 126.4|163.29|179.15|110.16| 128.6|151.92|153.78|142.69|146.6
147.62|141.28|138.34|134.31|143.02|156.51|150.22| 141.5|123.21| 139.7|113.19|130.56|154.02|163.95|159.2
177.47|176.36|146.81|142.19|165.17|180.51|161.85|143.72|160.17|185.64| 158.6|152.42|164.09|171.05|172.8
 147.6|162.96|136.81|133.87|151.56|175.62|195.69| 188.7|139.98|133.78|125.24|141.48|142.53|165.34|195.2
 126.3|121.82|121.28| 126.8|147.33| 140.8|123.66|138.44|142.33| 131.5|126.21|114.23|135.89|137.09|115.9
------+------+------+------+------+------+------+------+------+------+------+------+------+------+-----
```

In [0]:

```
from pyspark.sql.functions import rand
train_cw,test_cw = cw.randomSplit([.75,.25])
```

In [69]:

```
#shape of train and test
print(spark_shape(train_cw))
print(spark_shape(test_cw))
```

```
(1569, 65)
(479, 65)
```

Files can be converted to a pandas dataframe and saved as a csv or converted to a spark file to be saved

In [0]:
```
train_cw.toPandas().to_csv('/content/drive/My Drive/Colab IAF 604/train_carwood.csv')
```

In [0]:
```
train_cw.write.format("csv").save('/content/drive/My Drive/Colab IAF 604/train_carwood.csv')
```

In [0]:
```
test_cw.toPandas().to_csv('/content/drive/My Drive/Colab IAF 604/test_carwood.csv')
```

In [0]:
```
test_cw.write.format("csv").save('/content/drive/My Drive/Colab IAF 604/test_carwood.csv')
```