# banknote_data_analysis

February 2, 2019

In [239]:
```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
%matplotlib inline
```

#### 0.0.1 Banknote dataset

**Source:** http://archive.ics.uci.edu/ml/datasets/banknote+authentication

**Data Set Information:** Data were extracted from images that were taken from genuine and forged banknote-like specimens. For digitization, an industrial camera usually used for print inspection was used. The final images have 400x 400 pixels. Due to the object lens and distance to the investigated object gray-scale pictures with a resolution of about 660 dpi were gained. Wavelet Transform tool were used to extract features from images.

**Attribute Information:**

1. variance of Wavelet Transformed image (continuous)
2. skewness of Wavelet Transformed image (continuous)
3. curtosis of Wavelet Transformed image (continuous)
4. entropy of image (continuous)
5. class (Authentic(0) and Counterfeit(1))

In [240]:
```
file = 'data_banknote_authentication.txt'
df = pd.read_csv(file, names=['variance','skewness','kurtosis','entropy','class'])
df.shape
```

Out[240]: (1372, 5)

In [241]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1372 entries, 0 to 1371
Data columns (total 5 columns):
variance    1372 non-null float64
skewness    1372 non-null float64
kurtosis    1372 non-null float64
```

```
entropy     1372 non-null float64
class       1372 non-null int64
dtypes: float64(4), int64(1)
memory usage: 53.7 KB
```

In [242]:
```
df.head()
```

Out[242]:

| | variance | skewness | kurtosis | entropy | class |
|---|---|---|---|---|---|
| 0 | 3.62160 | 8.6661 | -2.8073 | -0.44699 | 0 |
| 1 | 4.54590 | 8.1674 | -2.4586 | -1.46210 | 0 |
| 2 | 3.86600 | -2.6383 | 1.9242 | 0.10645 | 0 |
| 3 | 3.45660 | 9.5228 | -4.0112 | -3.59440 | 0 |
| 4 | 0.32924 | -4.4552 | 4.5718 | -0.98880 | 0 |

In [243]:
```
df.describe()
```

Out[243]:

| | variance | skewness | kurtosis | entropy | class |
|---|---|---|---|---|---|
| count | 1372.000000 | 1372.000000 | 1372.000000 | 1372.000000 | 1372.000000 |
| mean | 0.433735 | 1.922353 | 1.397627 | -1.191657 | 0.444606 |
| std | 2.842763 | 5.869047 | 4.310030 | 2.101013 | 0.497103 |
| min | -7.042100 | -13.773100 | -5.286100 | -8.548200 | 0.000000 |
| 25% | -1.773000 | -1.708200 | -1.574975 | -2.413450 | 0.000000 |
| 50% | 0.496180 | 2.319650 | 0.616630 | -0.586650 | 0.000000 |
| 75% | 2.821475 | 6.814625 | 3.179250 | 0.394810 | 1.000000 |
| max | 6.824800 | 12.951600 | 17.927400 | 2.449500 | 1.000000 |

In [244]:
```
# check dataframe for bad data
df.isnull().any() | df.isna().any()
```

Out[244]:
```
variance    False
skewness    False
kurtosis    False
entropy     False
class       False
dtype: bool
```

In [245]:
```
df.isnull().sum()
```

Out[245]:
```
variance    0
skewness    0
kurtosis    0
entropy     0
class       0
dtype: int64
```

In [246]:
```
df.isna().sum()
```

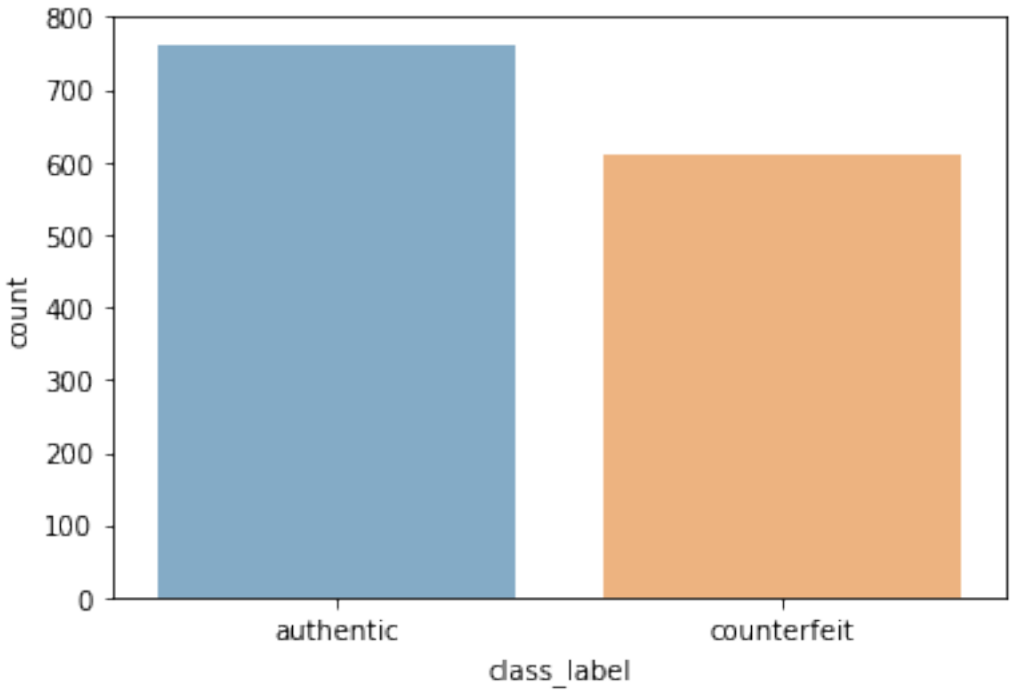
```
Out[246]: variance    0
          skewness    0
          kurtosis    0
          entropy     0
          class       0
          dtype: int64
```

```
In [247]: # add a class label based on the class value (to make labeling plots easier)
          df['class_label'] = np.where(df['class']>0, 'counterfeit', 'authentic')
```

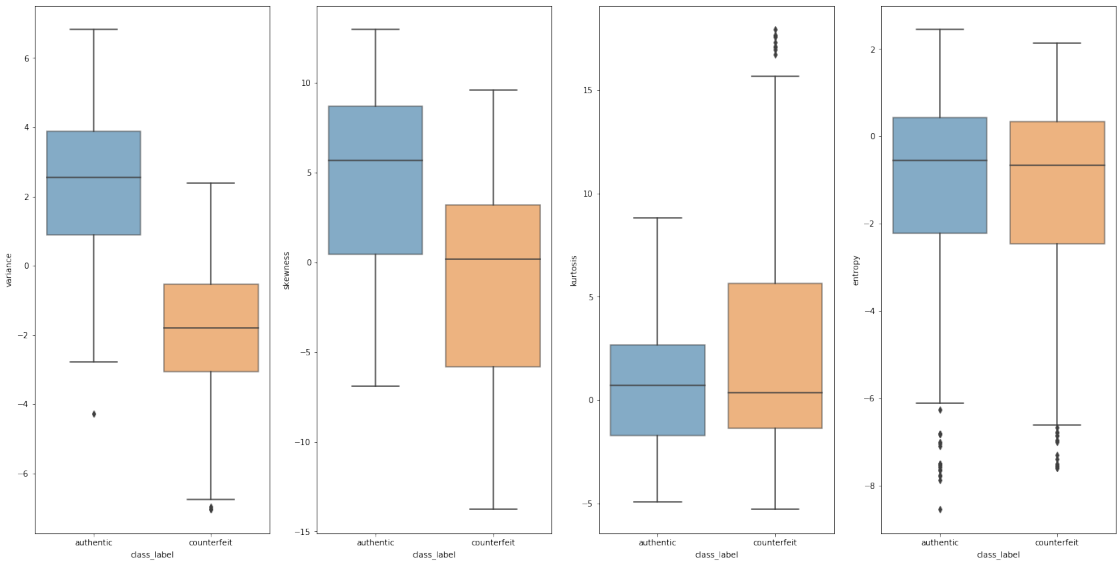
```
In [248]: # get the feature names to make plotting easier
          feature_names = list(df.columns.values)
          feature_names.remove('class')
          feature_names.remove('class_label')
```

```
In [263]: # get the number of samples for each class
          num_authentic = df[df['class'] == 0].shape[0]
          num_counterfeit = df[df['class'] == 1].shape[0]

          print('Authentic:    ',num_authentic)
          print('Counterfeit: ', num_counterfeit)

          sns_plot = sns.countplot(x='class_label', data=df, alpha=.6)
          plt.savefig('banknote_countplot.png')
          plt.show()
```

```
Authentic:    762
Counterfeit:  610
```

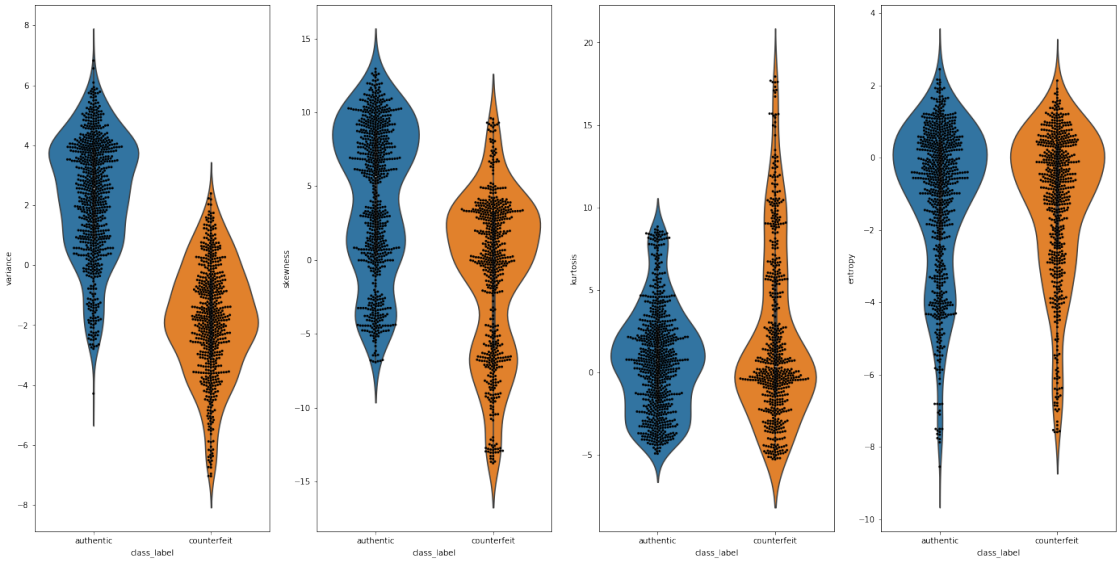


In [262]:
```
num_plots = len(feature_names)
num_columns = 4
num_rows = int(num_plots/num_columns + num_plots%num_columns) # round up

fig, axs = plt.subplots(nrows=num_rows, ncols=num_columns, figsize=(24,12))

if num_rows == 1:
    for index, name in enumerate(feature_names):
        sns.boxplot(x='class_label', y=name, data=df, ax=axs[index], boxprops=dict(alpha=.6))
else:
    for index, name in enumerate(feature_names):
        row = int(index / num_columns)
        col = index % num_columns
        sns.boxplot(x='class_label', y=name, data=df, ax=axs[row][col], boxprops=dict(alpha=.6))
plt.savefig('banknote_boxplots.png')
plt.show()
```

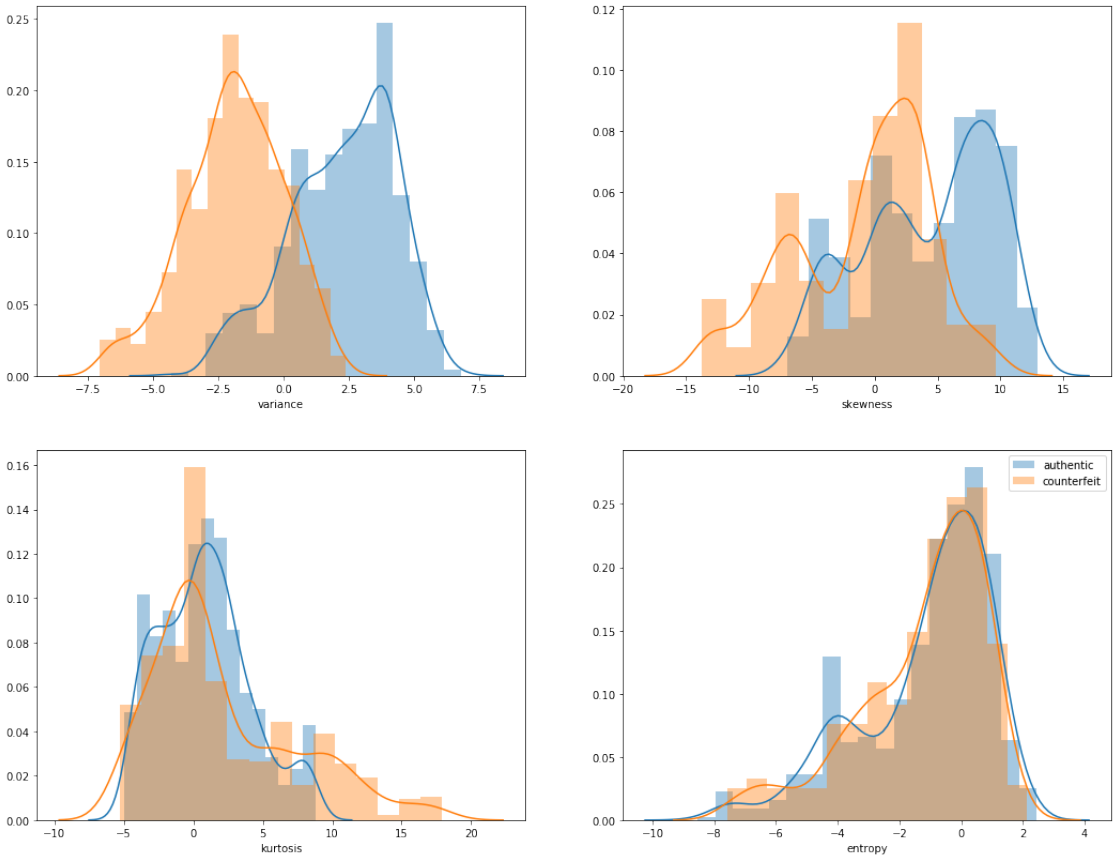



In [261]:
```
num_plots = len(feature_names)
num_columns = 4
num_rows = int(num_plots/num_columns + num_plots%num_columns) # round up

fig, axs = plt.subplots(nrows=num_rows, ncols=num_columns, figsize=(24,12))

if num_rows == 1:
    for index, name in enumerate(feature_names):
        sns.violinplot(x='class_label', y=name, data=df, ax=axs[index])
```

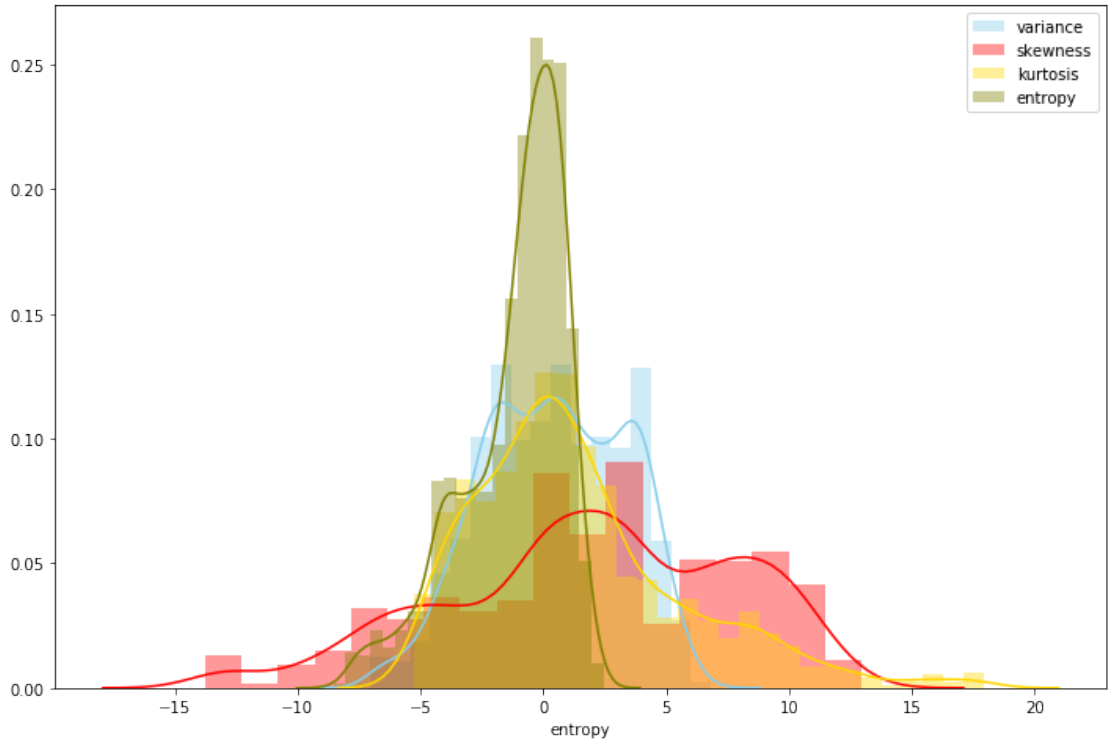
```
            sns.swarmplot(x='class_label', y=name, data=df, ax=axs[index], color='k', size=3);
    else:
        for index, name in enumerate(feature_names):
            row = int(index / num_columns)
            col = index % num_columns
            sns.violinplot(x='class_label', y=name, data=df, ax=axs[row][col])
            sns.swarmplot(x='class_label', y=name, data=df, ax=axs[row][col], color='k', size=3);
    plt.savefig('banknote_violinplots.png')
    plt.show()
```




In [260]:
```
num_plots = len(feature_names)
num_columns = 2
num_rows = int(num_plots/num_columns + num_plots%num_columns) # round up

fig, axs = plt.subplots(nrows=num_rows, ncols=num_columns, figsize=(18,14))

if num_rows == 1:
    for index, name in enumerate(feature_names):
        sns.distplot(df[df['class']==0][name], ax=axs[index])
        sns.distplot(df[df['class']==1][name], ax=axs[index])
else:
    for index, name in enumerate(feature_names):
        row = int(index / num_columns)
        col = index % num_columns
        sns.distplot(df[df['class']==0][name], ax=axs[row][col])
        sns.distplot(df[df['class']==1][name], ax=axs[row][col])
plt.legend(['authentic','counterfeit'])
plt.savefig('banknote_distplots_by_class.png')
plt.show()
```




In [253]:
```
# distribution plot without factoring in class
fig, axs = plt.subplots(figsize=(12,8))
sns.distplot(df['variance'], color='skyblue')
sns.distplot(df['skewness'], color='red')
sns.distplot(df['kurtosis'], color='gold')
sns.distplot(df['entropy'], color='olive')
plt.legend(['variance','skewness','kurtosis','entropy'])
plt.savefig('banknote_distplots.png')
plt.show()
```



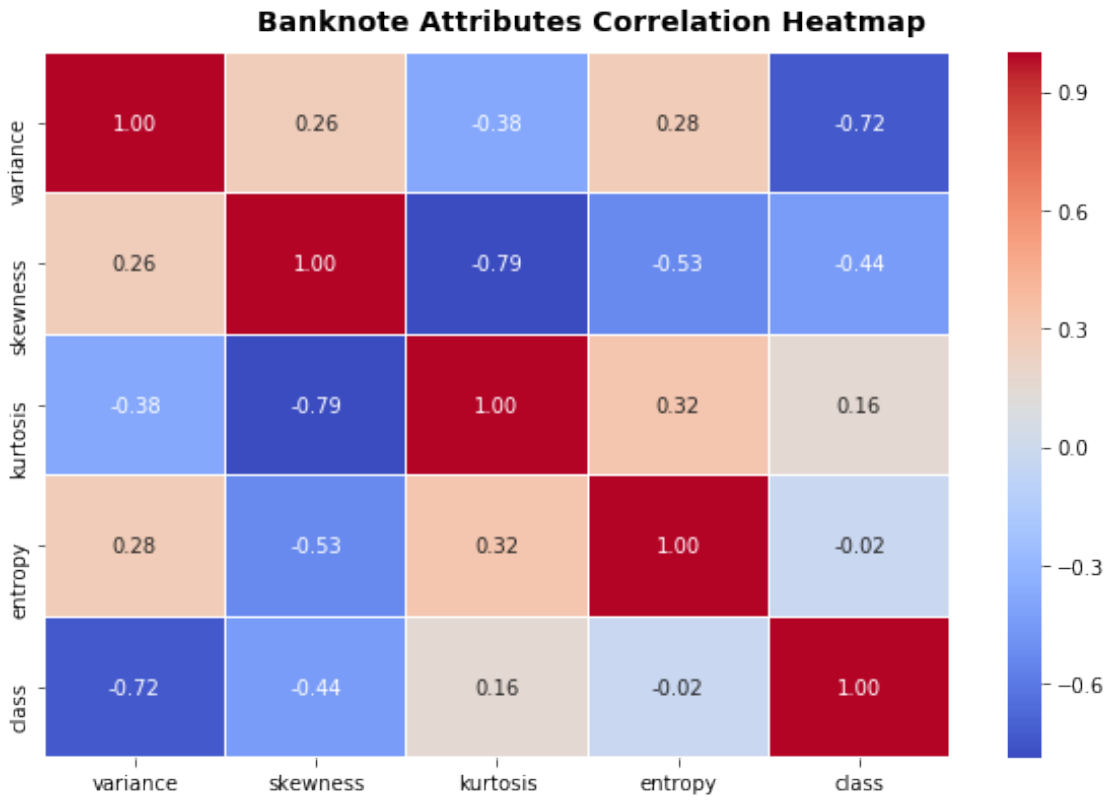

In [254]:
```
# compute pairwise correlation of the attributes
corr = df.corr()
corr
```

Out[254]:

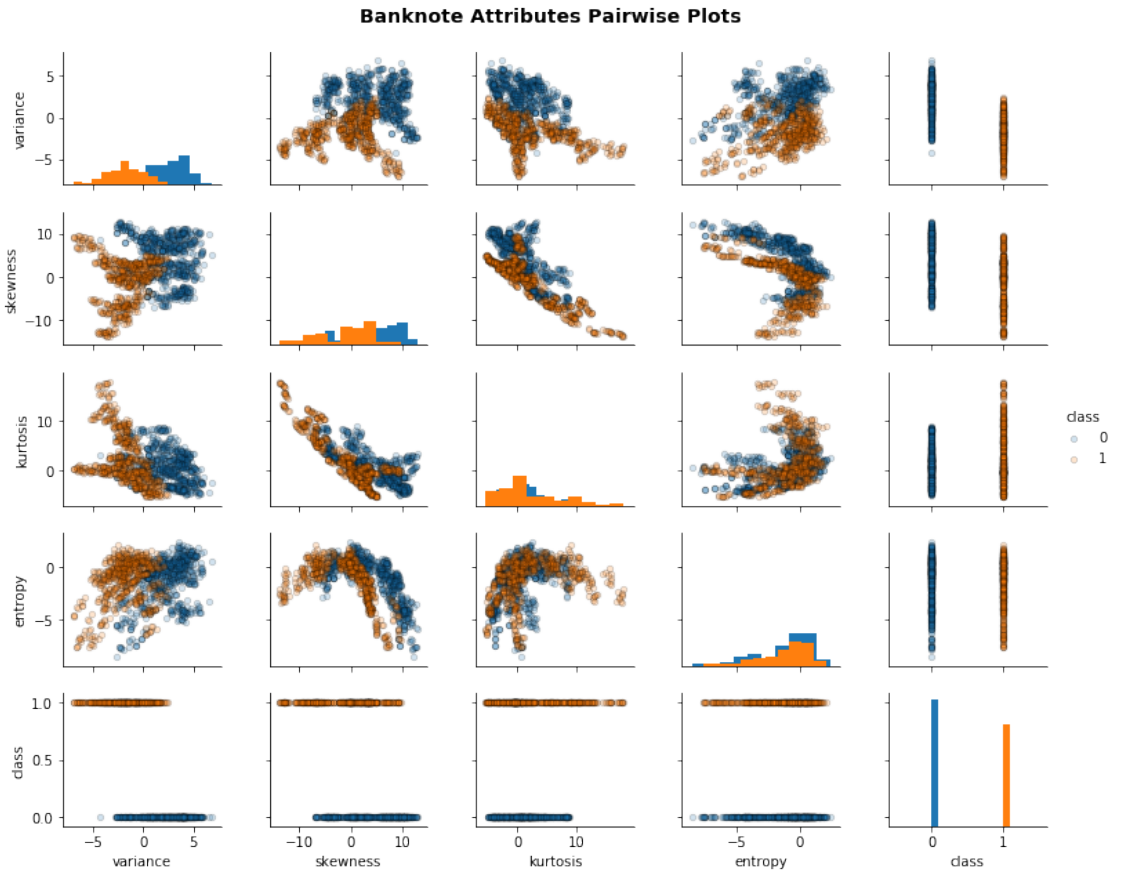
| | variance | skewness | kurtosis | entropy | class |
|---|---|---|---|---|---|
| variance | 1.000000 | 0.264026 | -0.380850 | 0.276817 | -0.724843 |
| skewness | 0.264026 | 1.000000 | -0.786895 | -0.526321 | -0.444688 |
| kurtosis | -0.380850 | -0.786895 | 1.000000 | 0.318841 | 0.155883 |
| entropy | 0.276817 | -0.526321 | 0.318841 | 1.000000 | -0.023424 |
| class | -0.724843 | -0.444688 | 0.155883 | -0.023424 | 1.000000 |

In [255]:
```
fig, (ax) = plt.subplots(1, 1, figsize=(10,6))

    hm = sns.heatmap(corr,
                     ax=ax,          # Axes in which to draw the plot, otherwise use the currently-active Axes.
                     cmap="coolwarm", # Color Map.
                     #square=True,    # If True, set the Axes aspect to equal so each cell will be square-shaped.
                     annot=True,
                     fmt='.2f',       # String formatting code to use when adding annotations.
                     #annot_kws={"size": 14},
                     linewidths=.05)

    fig.subplots_adjust(top=0.93)
```

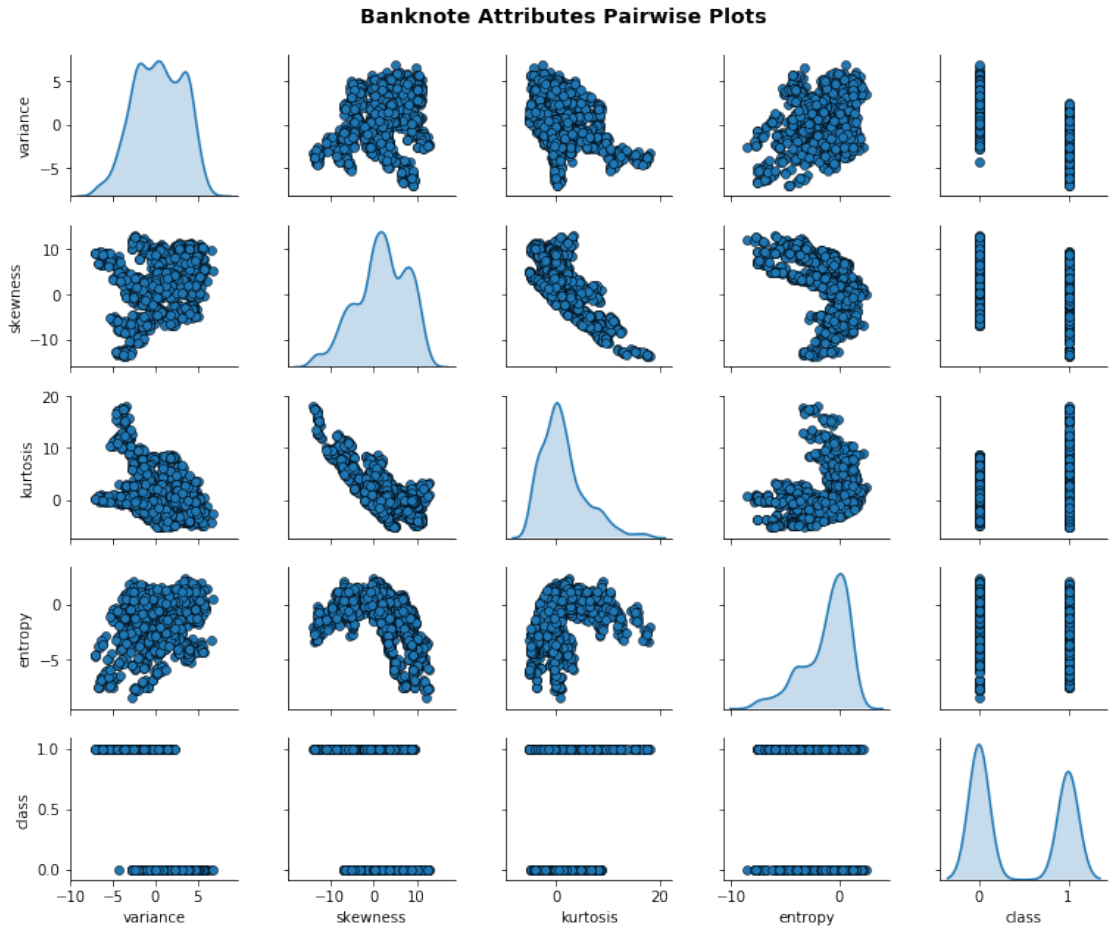
```
fig.suptitle('Banknote Attributes Correlation Heatmap', fontsize=14, fontweight='bold')
plt.savefig('banknote_heatmap.png')
plt.show()
```



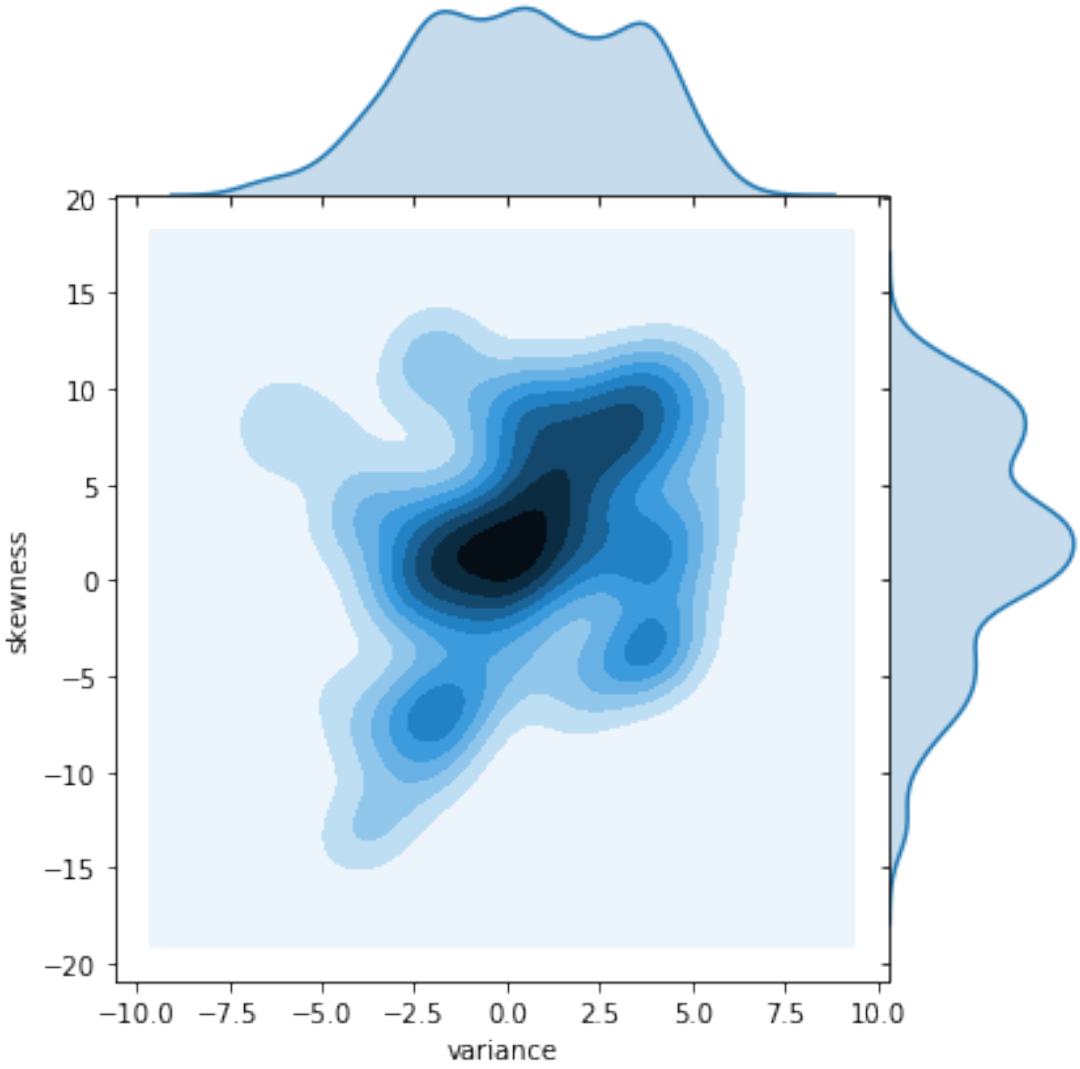

In [258]:
```
pp = sns.pairplot(df, height=1.8, aspect=1.2, hue='class', diag_kind='hist',
                  plot_kws = {'alpha': 0.2, 's': 20, 'edgecolor': 'k'})

fig = pp.fig
fig.subplots_adjust(top=0.93, wspace=0.3)
fig.suptitle('Banknote Attributes Pairwise Plots', fontsize=14, fontweight='bold')
plt.savefig('banknote_pairplot_hist.png')
plt.show()
```



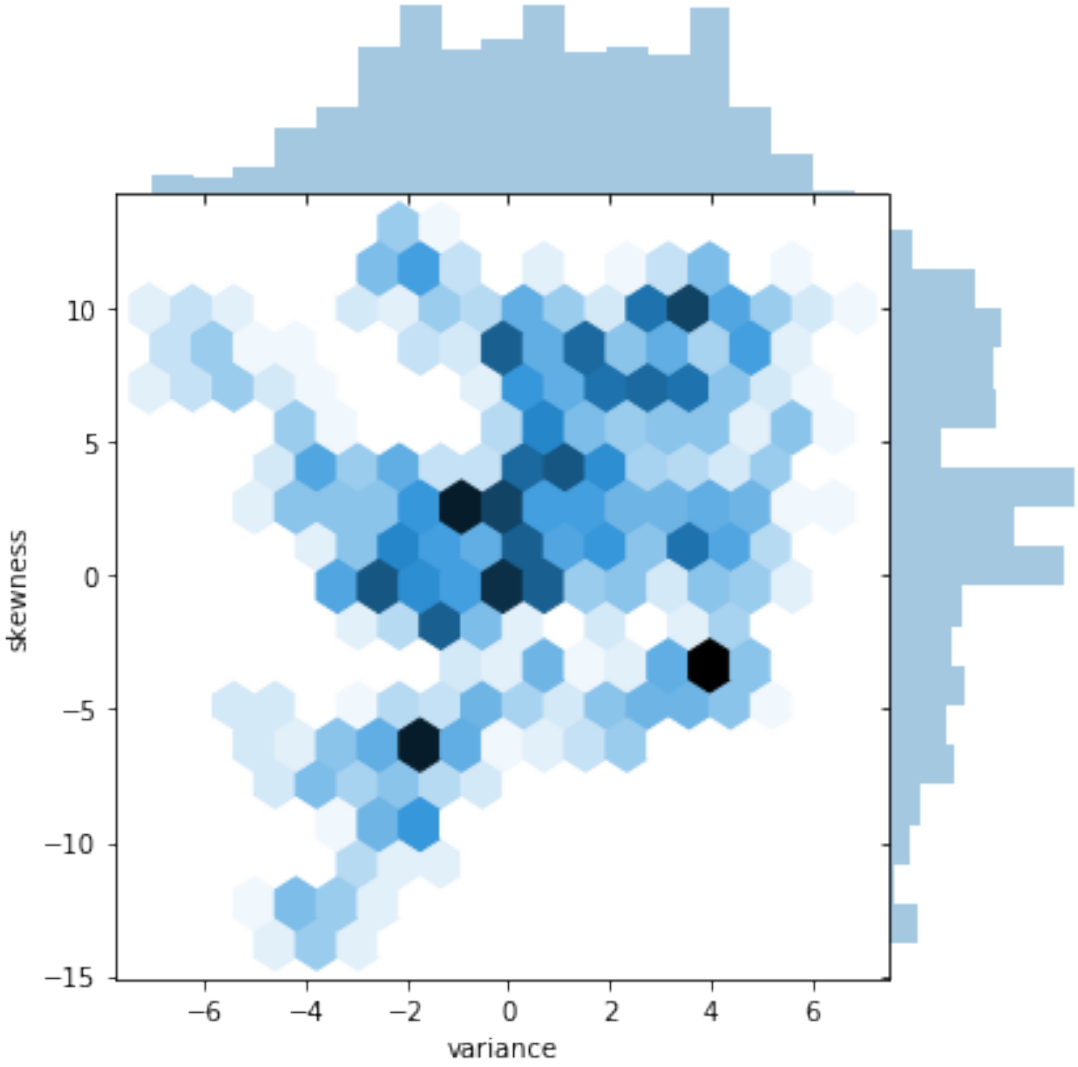

```
In [257]: pp = sns.pairplot(df, height=1.8, aspect=1.2,
                            plot_kws=dict(edgecolor="k", linewidth=0.5),
                            diag_kws=dict(shade=True), # "diag" adjusts/tunes the diagonal plots
                            diag_kind="kde") # use "kde" for diagonal plots

          fig = pp.fig
          fig.subplots_adjust(top=0.93, wspace=0.3)
          fig.suptitle('Banknote Attributes Pairwise Plots', fontsize=14, fontweight='bold')
          plt.savefig('banknote_pairplot.png')
          plt.show()
```



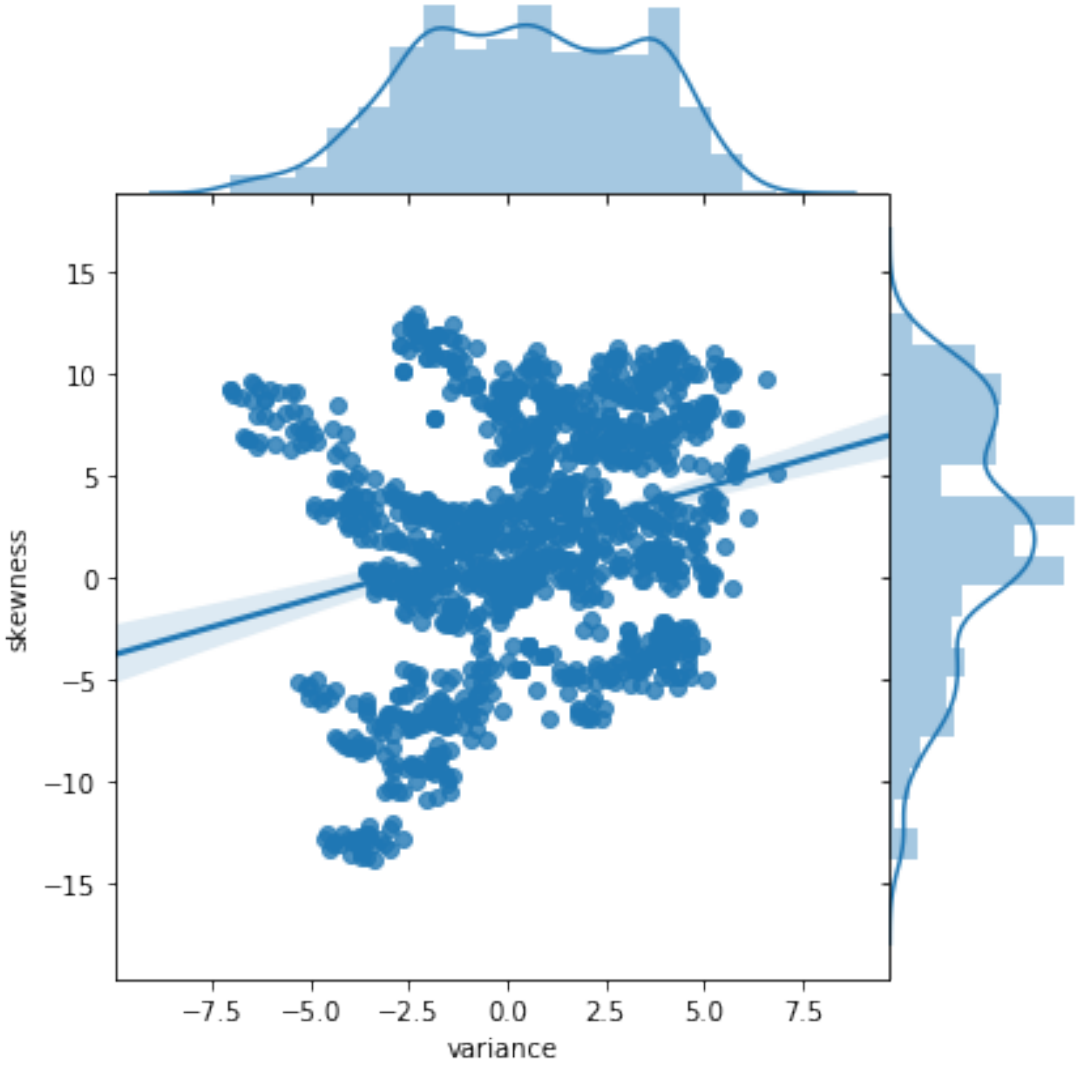

```
In [151]: jp = sns.jointplot(data=df,
                             x='variance',
                             y='skewness',
                             kind='kde',
                             space=0, height=6, ratio=4)
```




In [150]:
```
jp = sns.jointplot(data=df,
                   x='variance',
                   y='skewness',
                   kind='hex',
                   space=0, height=6, ratio=4)
```




In [152]:
```
jp = sns.jointplot(data=df,
                   x='variance',
                   y='skewness',
                   kind='reg',
                   space=0, height=6, ratio=4)
```


skewness
variance
15
10
5
0
−5
−10
−15
−7.5
−5.0
−2.5
0.0
2.5
5.0
7.5