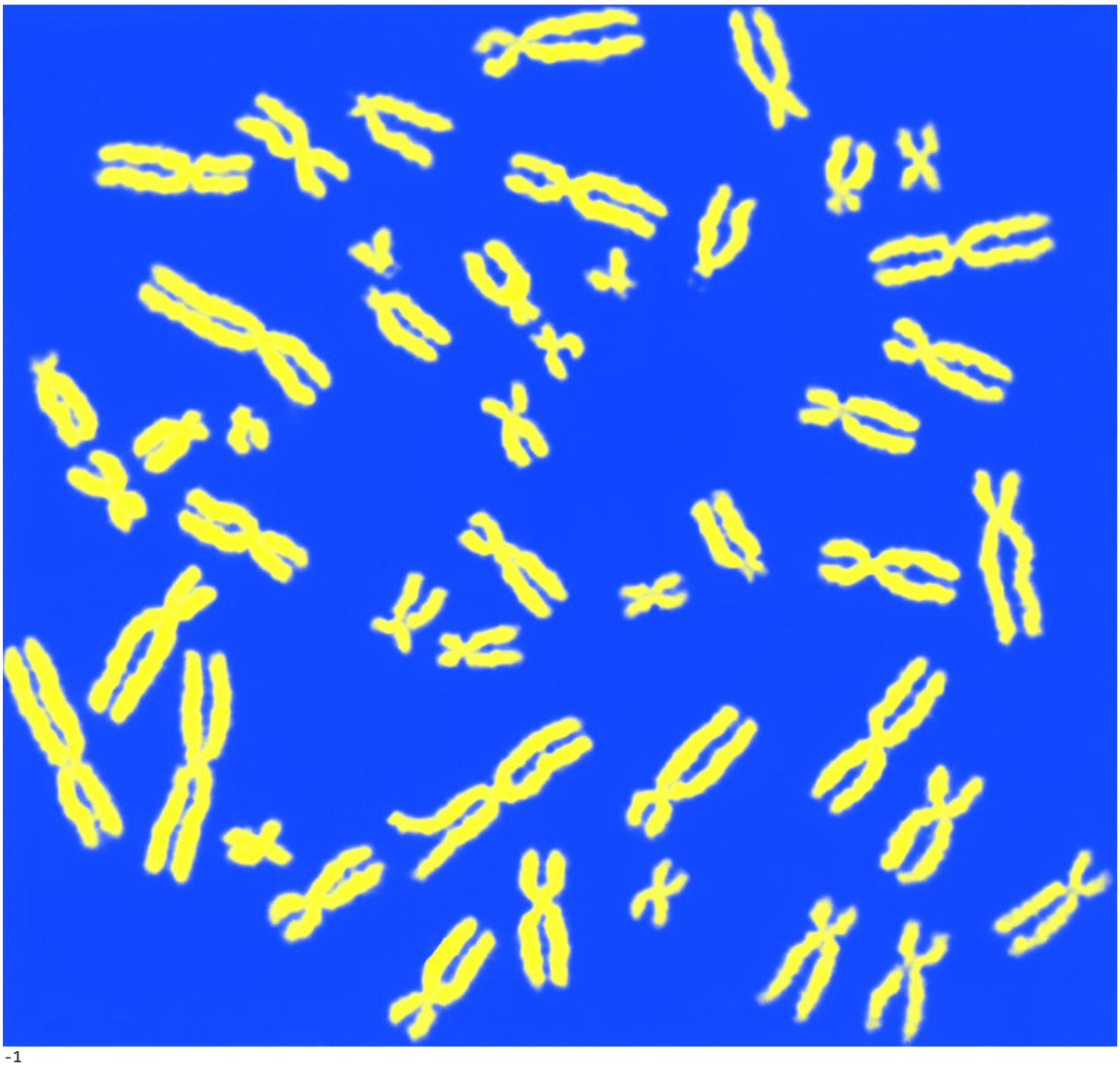
```
import cv2
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from pandas.plotting import table
from google.colab.patches import cv2_imshow
from sklearn.preprocessing import StandardScaler
```

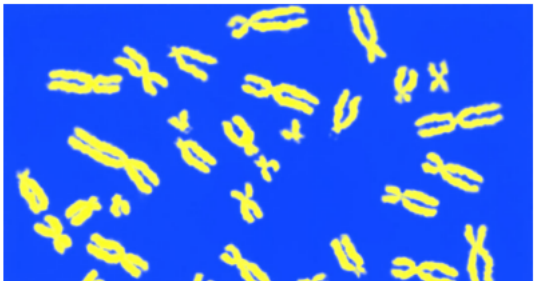
**Read the image file and plot it.**

```
# Load the input image
image = cv2.imread('chromosomes.jpg')
cv2_imshow(image)
cv2.waitKey(0)
```



```
-1
```

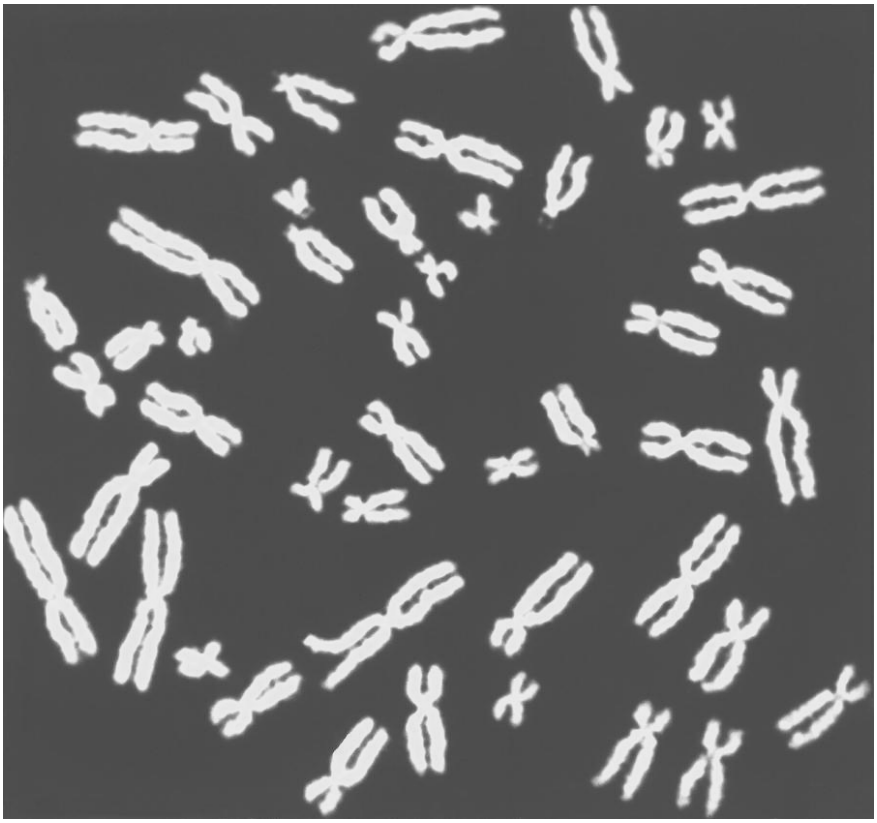
```
# Plot the input image

image_rgb = cv2.cvtColor(image, cv2.COLOR_BGR2RGB)
fig = plt.gcf()
plt.imshow(image_rgb)
plt.axis('off')
plt.show()
```



**Convert the image into GRAYSCALE**

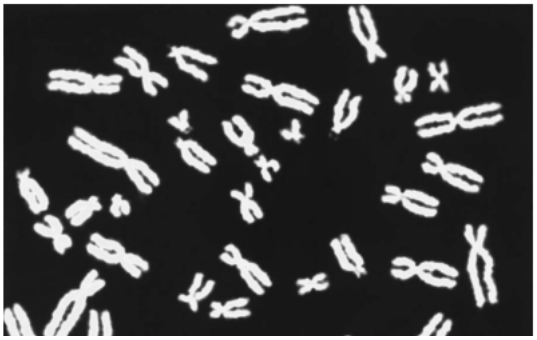


```
# Use the cvtColor() function to convert the image into grayscale from BGR
gray_image = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)

cv2_imshow(gray_image)
cv2.waitKey(0)
```



```
-1
```

```
# Plot the grayscale image

fig = plt.gcf()
plt.imshow(gray_image, cmap='gray')
plt.axis('off')
plt.show()
```



**Morphological opening** - to remove the blots in the background

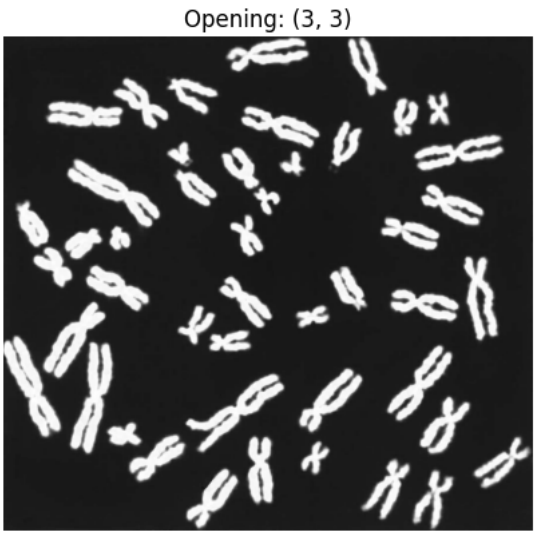
---

An opening is an erosion followed by a dilation.

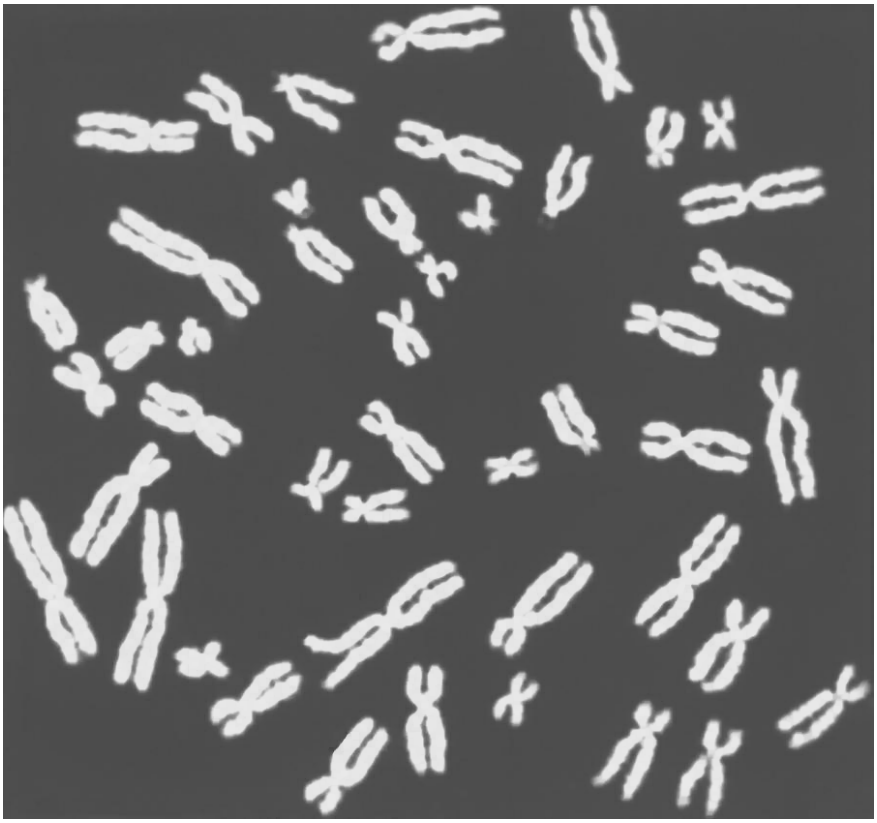
First an erosion is applied to remove the small blobs, then a dilation is applied to regrow the size of the original object.

---

```
kernelSizes = [(3, 3), (5, 5), (7, 7)]

for kernelSize in kernelSizes:
    # Construct a rectangular kernel from the current size and then apply an "opening" operation
  kernel = cv2.getStructuringElement(cv2.MORPH_RECT, kernelSize)
  opening = cv2.morphologyEx(gray_image, cv2.MORPH_OPEN, kernel)

  fig = plt.gcf()
  plt.imshow(opening, cmap='gray')
  plt.title('Opening: ({}, {})'.format(kernelSize[0], kernelSize[1]))
  plt.axis('off')
  plt.show()
  print('\n')
```

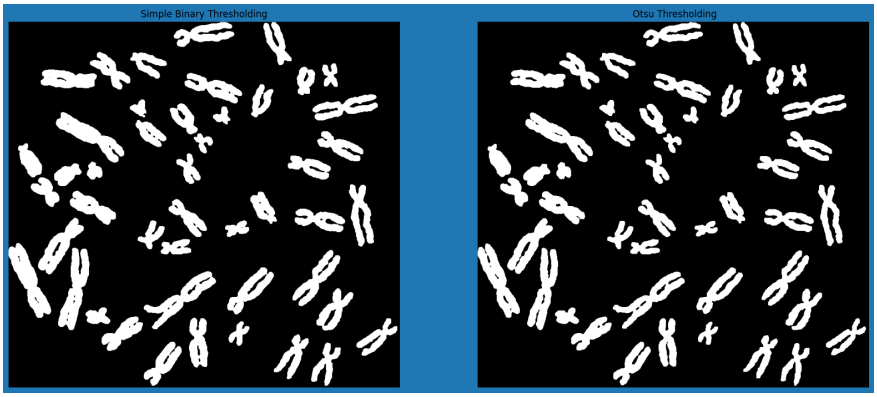



```
kernel = cv2.getStructuringElement( cv2.MORPH_RECT, (5, 5) )
img_open = cv2.morphologyEx(gray_image, cv2. MORPH_OPEN, kernel)
cv2_imshow(img_open)
```



**Threshold the image for binarization**

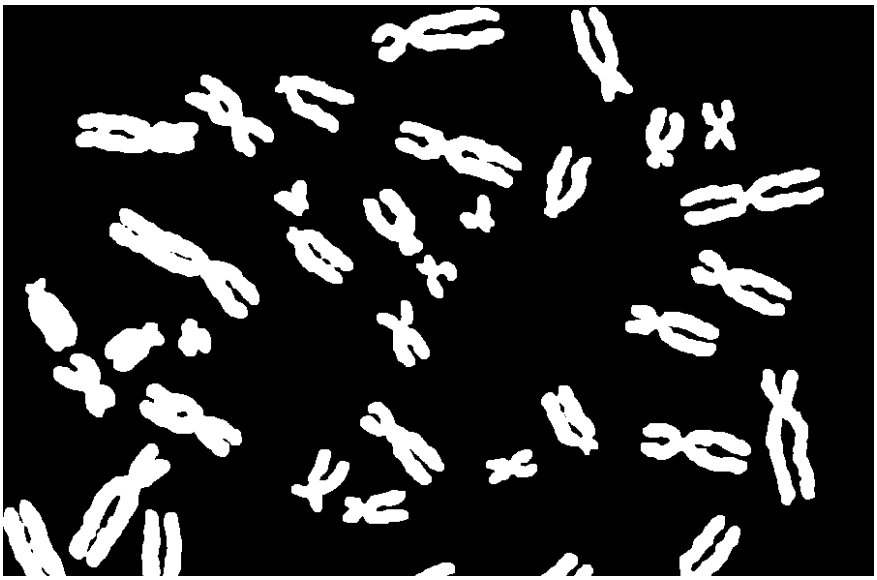
```
# Apply different thresholding techniques to find the most suitable one for the image

fig, (ax1, ax2) = plt.subplots(1, 2, sharey=True, figsize=(20, 20))

T, thresh_image1 = cv2.threshold(img_open, 127, 255, cv2.THRESH_BINARY)
ax1.imshow(thresh_image1, cmap='gray')
ax1.set_title('Simple Binary Thresholding')
ax1.axis('off')

T, thresh_image2 = cv2.threshold(img_open, 0, 255, cv2.THRESH_OTSU)
ax2.imshow(thresh_image2, cmap='gray')
ax2.set_title(' Otsu Thresholding')
ax2.axis('off')

fig.set_facecolor(color=None)
plt.show()
```




**Observation** - The segmentation by binarization using Otsu thresholding has more distinct result compared to the output of Simple binary thresholding.

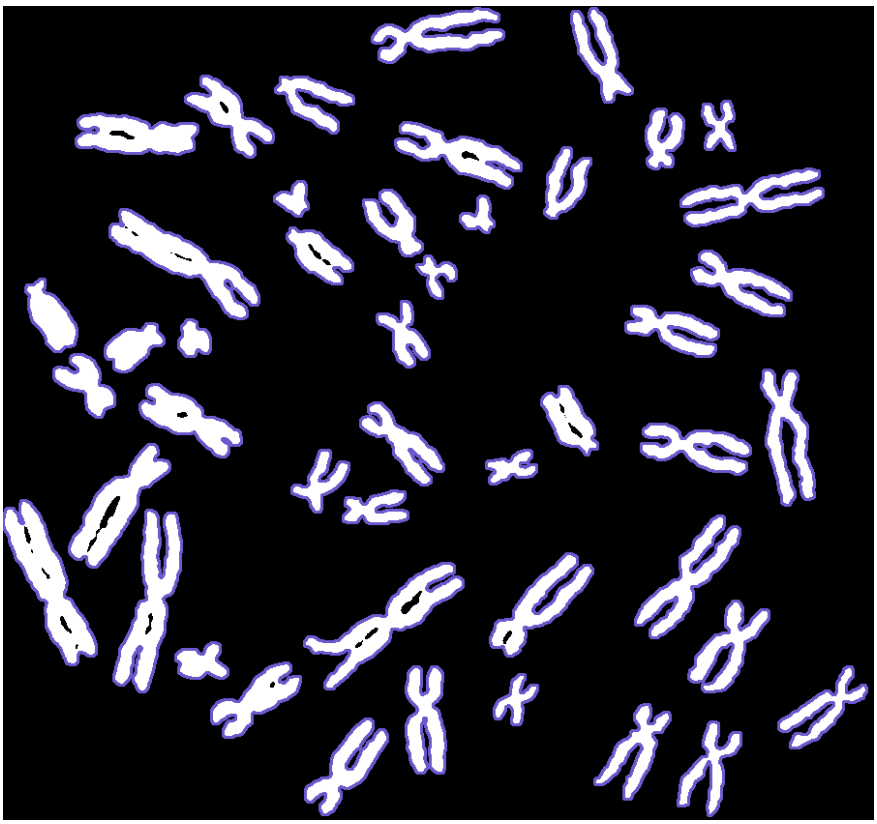
```
T, thresh_image = cv2.threshold(img_open, 0, 255, cv2.THRESH_OTSU)
print(f'Threshold value T = {T}\n')
cv2_imshow(thresh_image)
```

```
Threshold value T = 149.0
```



**Find the contours**

```
thresh_copy = thresh_image.copy()
contours, hierarchy = cv2.findContours(thresh_image.copy(), cv2.RETR_EXTERNAL, cv2.CHAIN_APPROX_NONE)

img_contour = cv2.cvtColor(thresh_copy, cv2.COLOR_GRAY2BGR)
cv2.drawContours(img_contour, contours, -1, (205, 90, 106), 2)
cv2_imshow(img_contour)
cv2.waitKey(0)
```



```
-1
```

```
n_countours = len(contours)
print("Number of Contours found = " + str(n_countours))
```

```
Number of Contours found = 46
```

**Find the features for each chromosome**

```
area = []
perimeter = []
circularity = []
top_left_X = []
top_left_Y = []
height = []
width = []
shape = []

for contour in contours:
  area_contour = cv2.contourArea(contour)
  area.append(area_contour)
  perimeter_contour = cv2.arcLength(contour, True)
  perimeter.append(perimeter_contour)
  circularity.append((4 * np.pi * area_contour) / (perimeter_contour ** 2))

  X, Y, w, h = cv2.boundingRect(contour)
  top_left_X.append(X)
  top_left_Y.append(Y)
  width.append(w)
  height.append(h)
  shape.append(f'{w} x {h}')


chromosome_df = pd.DataFrame({'X': top_left_X,'Y': top_left_Y, 'Width': width, 'Height': height, 'shape_info': shape, 'area': area, 'per
print(f'Shape of the Chromosome dataframe = {chromosome_df.shape}\n')
chromosome_df
```

Shape of the Chromosome dataframe = (46, 8)

| | X | Y | Width | Height | shape_info | area | perimeter | circularity |
|---|---|---|---|---|---|---|---|---|
| **0** | 697 | 740 | 66 | 98 | 66 x 98 | 2172.5 | 415.806130 | 0.157902 |
| **1** | 312 | 737 | 86 | 100 | 86 x 100 | 3092.0 | 440.558437 | 0.200190 |
| **2** | 611 | 722 | 79 | 99 | 79 x 99 | 2515.5 | 431.203098 | 0.170008 |
| **3** | 507 | 691 | 46 | 53 | 46 x 53 | 1000.0 | 213.338094 | 0.276104 |
| **4** | 801 | 683 | 92 | 85 | 92 x 85 | 2285.0 | 438.901583 | 0.149060 |
| **5** | 416 | 683 | 42 | 111 | 42 x 111 | 3206.5 | 475.462984 | 0.178241 |
| **6** | 215 | 679 | 93 | 78 | 93 x 78 | 2957.0 | 354.534052 | 0.295628 |
| **7** | 178 | 658 | 55 | 38 | 55 x 38 | 1324.5 | 180.267026 | 0.512188 |
| **8** | 708 | 615 | 84 | 95 | 84 x 95 | 2912.0 | 460.759447 | 0.172366 |
| **9** | 311 | 576 | 165 | 131 | 165 x 131 | 5432.0 | 612.582822 | 0.181903 |
| **10** | 503 | 566 | 107 | 107 | 107 x 107 | 3680.0 | 497.470124 | 0.186863 |
| **11** | 653 | 527 | 109 | 127 | 109 x 127 | 3915.0 | 605.553385 | 0.134164 |
| **12** | 114 | 521 | 71 | 189 | 71 x 189 | 5723.0 | 690.759447 | 0.150723 |
| **13** | 0 | 511 | 97 | 171 | 97 x 171 | 6009.5 | 489.830515 | 0.314743 |
| **14** | 351 | 501 | 68 | 35 | 68 x 35 | 1500.5 | 265.580734 | 0.267333 |
| **15** | 499 | 459 | 53 | 36 | 53 x 36 | 1003.5 | 193.438599 | 0.337009 |
| **16** | 298 | 459 | 60 | 65 | 60 x 65 | 1475.0 | 265.622364 | 0.262708 |
| **17** | 68 | 453 | 105 | 128 | 105 x 128 | 4794.5 | 386.031525 | 0.404304 |
| **18** | 659 | 432 | 114 | 53 | 114 x 53 | 3008.0 | 485.989895 | 0.160042 |

```
ax = plt.subplot(111, frame_on=False) # no visible frame
ax.xaxis.set_visible(False)  # hide the x axis
ax.yaxis.set_visible(False)  # hide the y axis

df_table = table(ax, chromosome_df)
ax.set_frame_on(False)
df_table.auto_set_column_width(range(len(chromosome_df.columns)))
df_table.auto_set_font_size(True)
df_table.scale(1.2, 1.2)
plt.savefig('Output_Chromosome_DataFrame.png', format = 'png')
```

| | X | Y | Width | Height | shape_info | area | perimeter | circularity |
|---|---|---|---|---|---|---|---|---|
| 0 | 697 | 740 | 66 | 98 | 66 x 98 | 2172.5 | 415.8061298131943 | 0.15790208852046916 |
| 1 | 312 | 737 | 86 | 100 | 86 x 100 | 3092.0 | 440.55843687057495 | 0.20018996522331636 |
| 2 | 611 | 722 | 79 | 99 | 79 x 99 | 2515.5 | 431.20309841632843 | 0.17000842346411268 |
| 3 | 507 | 691 | 46 | 53 | 46 x 53 | 1000.0 | 213.33809351921082 | 0.27610422004998747 |
| 4 | 801 | 683 | 92 | 85 | 92 x 85 | 2285.0 | 438.9015827178955 | 0.14906022571398572 |
| 5 | 416 | 683 | 42 | 111 | 42 x 111 | 3206.5 | 475.4629839658737 | 0.17824103076298595 |
| 6 | 215 | 679 | 93 | 78 | 93 x 78 | 2957.0 | 354.534051656723 | 0.29562780636403463 |
| 7 | 178 | 658 | 55 | 38 | 55 x 38 | 1324.5 | 180.2670258283615 | 0.5121878118331447 |
| 8 | 708 | 615 | 84 | 95 | 84 x 95 | 2912.0 | 460.7594473361969 | 0.17236645006075368 |
| 9 | 311 | 576 | 165 | 131 | 165 x 131 | 5432.0 | 612.5828220844269 | 0.18190305660988768 |
| 10 | 503 | 566 | 107 | 107 | 107 x 107 | 3680.0 | 497.47012424468994 | 0.18686315379318677 |
| 11 | 653 | 527 | 109 | 127 | 109 x 127 | 3915.0 | 605.5533845424652 | 0.1341642350352647 |
| 12 | 114 | 521 | 71 | 189 | 71 x 189 | 5723.0 | 690.7594473361969 | 0.1507231434885514 |
| 13 | 0 | 511 | 97 | 171 | 97 x 171 | 6009.5 | 489.8305150270462 | 0.31474332588393844 |
| 14 | 351 | 501 | 68 | 35 | 68 x 35 | 1500.5 | 265.5807341337204 | 0.26733309700677543 |
| 15 | 499 | 459 | 53 | 36 | 53 x 36 | 1003.5 | 193.4385987520218 | 0.3370085861382585 |
| 16 | 298 | 459 | 60 | 65 | 60 x 65 | 1475.0 | 265.62236428260803 | 0.26270758272859834 |

```
list(range(len(chromosome_df.columns)))
```

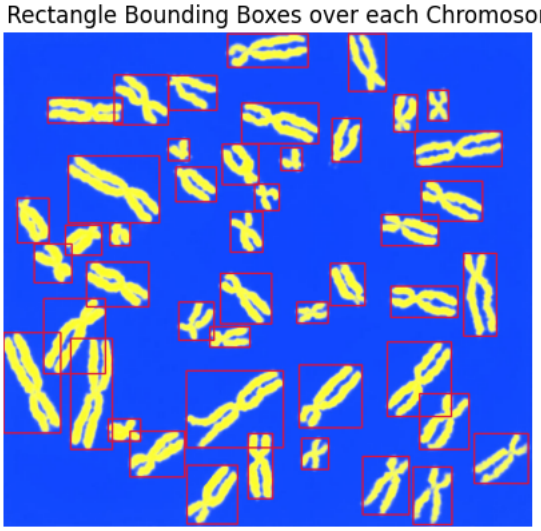
```
[0, 1, 2, 3, 4, 5, 6, 7]
```

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| 21 | 141 | 391 | 106 | 76 | 106 x 76 | 3384.0 | 343.7056245803833 | 0.35997052375578525 |

**Draw a bounding box for each chromosome**

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| 24 | 105 | 328 | 62 | 51 | 62 x 51 | 1778.0 | 187.68123936653137 | 0.6343075712960804 |

```
img = image.copy()

for idx in range(n_countours):
  X = top_left_X[idx]
  Y = top_left_Y[idx]
  h = height[idx]
  w = width[idx]

  cv2.rectangle(img,(X, Y),(X + w, Y + h), (60, 20, 220), 2)

img_rgb = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)
fig = plt.gcf()
plt.axis('off')
plt.title('Straight Rectangle Bounding Boxes over each Chromosome Pair')
plt.imshow(img_rgb)
plt.savefig('Straight_Rectangle_Bounding_Boxes.png', format = 'png')
```



**Normalise all the features using z-scoring**

```
scaler = StandardScaler().set_output(transform="pandas")

chr_df = chromosome_df.drop(['shape_info'], axis=1)
scaled_chromosome_df = scaler.fit_transform(chr_df)
scaled_chromosome_df
```

| | X | Y | Width | Height | area | perimeter | circularity |
|---|---|---|---|---|---|---|---|
| **0** | 1.299555 | 1.711050 | -0.414485 | 0.553956 | -0.346347 | 0.355871 | -0.914526 |
| **1** | -0.380994 | 1.697105 | 0.182270 | 0.611283 | 0.342358 | 0.524524 | -0.630160 |

```
def detect_outliers_zscore(data):
  outliers = []
  thres = 3
  mean = np.mean(data)
  std = np.std(data)
  for i in data:
      z_score = (i-mean)/std
      if (np.abs(z_score) > thres):
          outliers.append(i)
  return outliers
```

```
cols = chr_df.columns

for col in cols:
  sample_outliers = detect_outliers_zscore(chromosome_df[col])
  print(f'Outliers from Z-scores method in {col} = {sample_outliers}')
```

```
Outliers from Z-scores method in X = []
Outliers from Z-scores method in Y = []
Outliers from Z-scores method in Width = []
Outliers from Z-scores method in Height = [189]
Outliers from Z-scores method in area = []
Outliers from Z-scores method in perimeter = []
Outliers from Z-scores method in circularity = []
```

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| **17** | -1.446070 | 0.377011 | 0.749187 | 1.413864 | 1.617529 | 0.152997 | 0.742411 |
| **18** | 1.133683 | 0.279398 | 1.017727 | -0.735907 | 0.279442 | 0.834078 | -0.900138 |
| **19** | -0.132186 | 0.177138 | 0.212108 | 0.209992 | -0.075209 | 0.158642 | -0.545287 |
| **20** | 0.684081 | 0.098118 | -0.623350 | -0.191298 | -0.309272 | -0.943280 | 1.728095 |
| **21** | -1.127420 | 0.088821 | 0.779025 | -0.076644 | 0.561066 | -0.135396 | 0.444292 |
| **22** | 1.674951 | 0.019098 | -0.712863 | 1.786491 | 0.736332 | 1.585515 | -1.116443 |
| **23** | -1.515911 | -0.055273 | -0.474161 | -0.363280 | -0.499517 | -0.831924 | 0.875556 |
| **24** | -1.284562 | -0.204016 | -0.533836 | -0.793234 | -0.641827 | -1.198491 | 2.289083 |
| **25** | -0.952817 | -0.222609 | -1.369294 | -1.137198 | -1.305441 | -1.601264 | 2.583688 |
| **26** | 1.063842 | -0.287684 | 0.510485 | -0.735907 | -0.107042 | 0.229234 | -0.641726 |