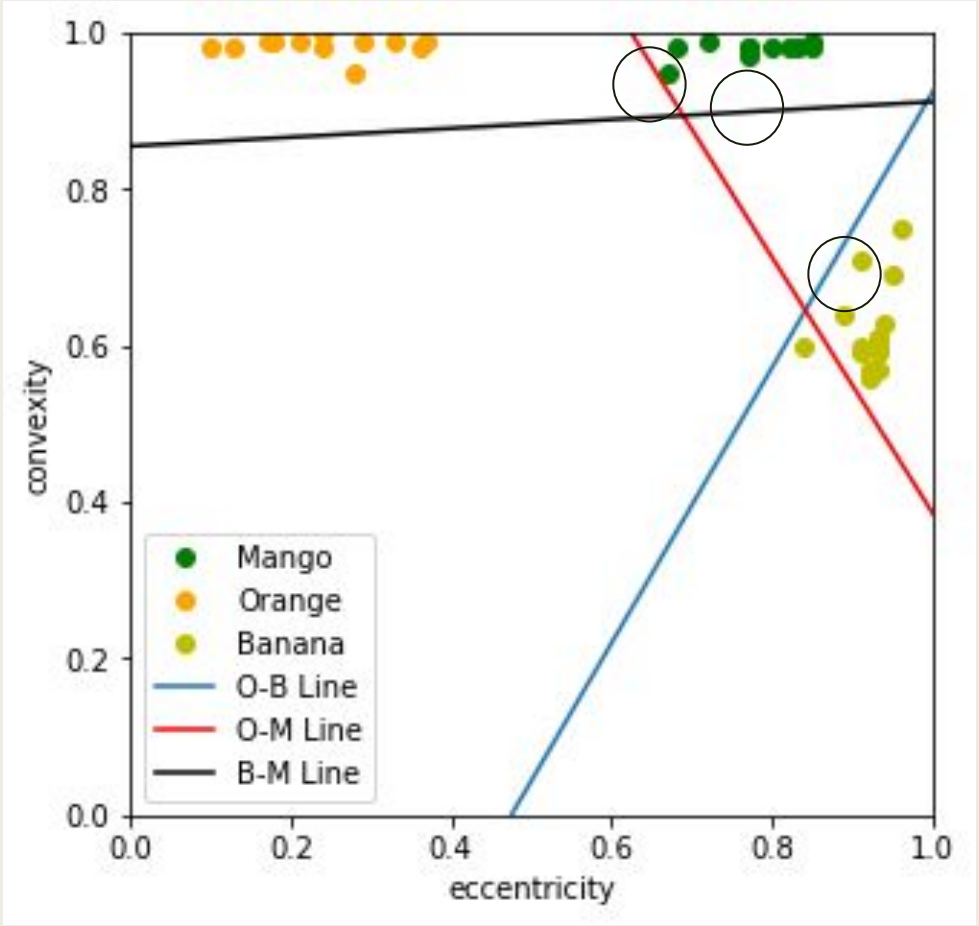
Support Vector Machines

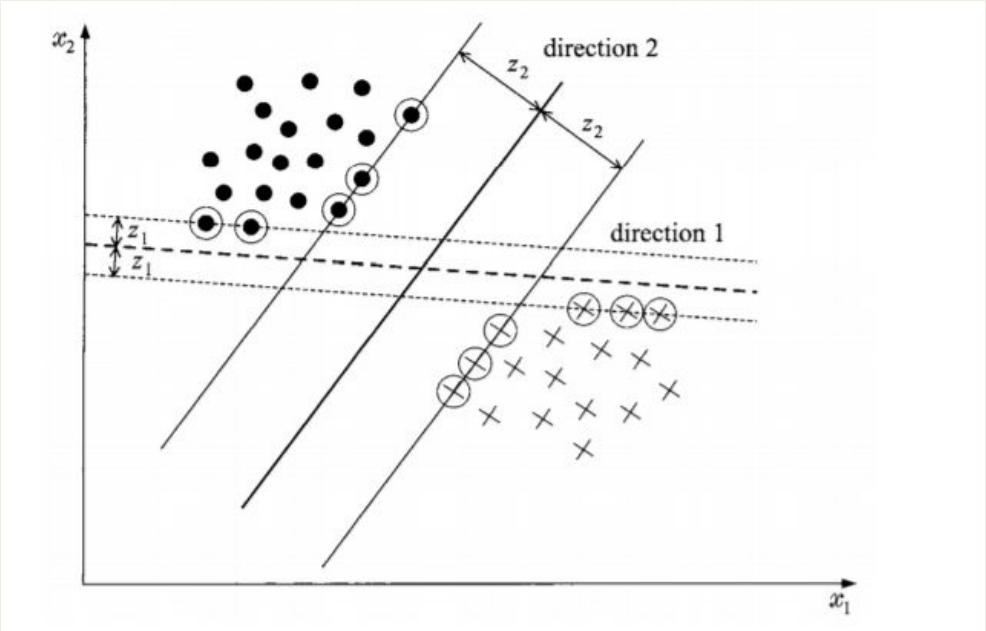
Objective

- In perceptron algorithm, the decision line is not optimal to split between 2 clusters thus we use support vector machines to find the optimal decision to split between 2 clusters.



Overview: Support Vector Machines

- In perceptron algorithm, the decision line is not optimal to split between 2 clusters.
- In support vector machines to find the optimal decision to split between 2 clusters using points the are close to the expected boundaries as shown in this image.
- We wish to maximize the margin z_1 or z_2.



In this case there are two possible directions that the decision line can optimally split either horizontally looking at points close to the horizontal boundary or vertically with points close to the ‘vertical’ boundary.

Overview: Optimal Hyperplane

- Using the equation shown in the image we wish to maximize the margin using a vector **α.**
- Where **α** is the optimized solution, **z_i** are the values that identify what cluster a sample belongs and **x_i** are the sample values.
- The equations can be fitted in to this form [bottom image] for quadratic programming where for **N** samples **H, A** and **B** are NxN matrices while **f, a** and **b** are N length vectors.

$$\text{maximize} \quad L_D(\alpha) = \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j z_i z_j x_i^t x_j$$

$$\text{constrained to} \quad \alpha_i \geq 0 \quad \forall i \quad \text{and} \quad \sum_{i=1}^{n} \alpha_i z_i = 0$$

Quadratic fxn the find a solution for **α**

$$\text{minimize} \quad L_D(\alpha) = 0.5\alpha^t H \alpha + f^t \alpha$$

$$\text{constrained to} \quad A\alpha \leq a \text{ and } B\alpha = b$$

Equivalent eqns. for quadractic programming

Step 0: Initializing variables

- Data used: fruits data (bananas, mangoes oranges) using their eccentricity and convexity features.
- **x** is a matrix the piles 2 fruits data with their respective features in a Nx2 array.
- **z** is a vector that collects all **z_i** values (ex. 1 for bananas, -1 for mangoes)
- **H = (x*x').*(z*z')**
- **f** is a N length vector with all values of -1
- **a** and **b** are N length vectors with all values of 0
- **A** is a NxN identity matrix
- **B** is a NxN matrix where the first row is the vector **z** while the rest are 0.

Step 1: Using the α

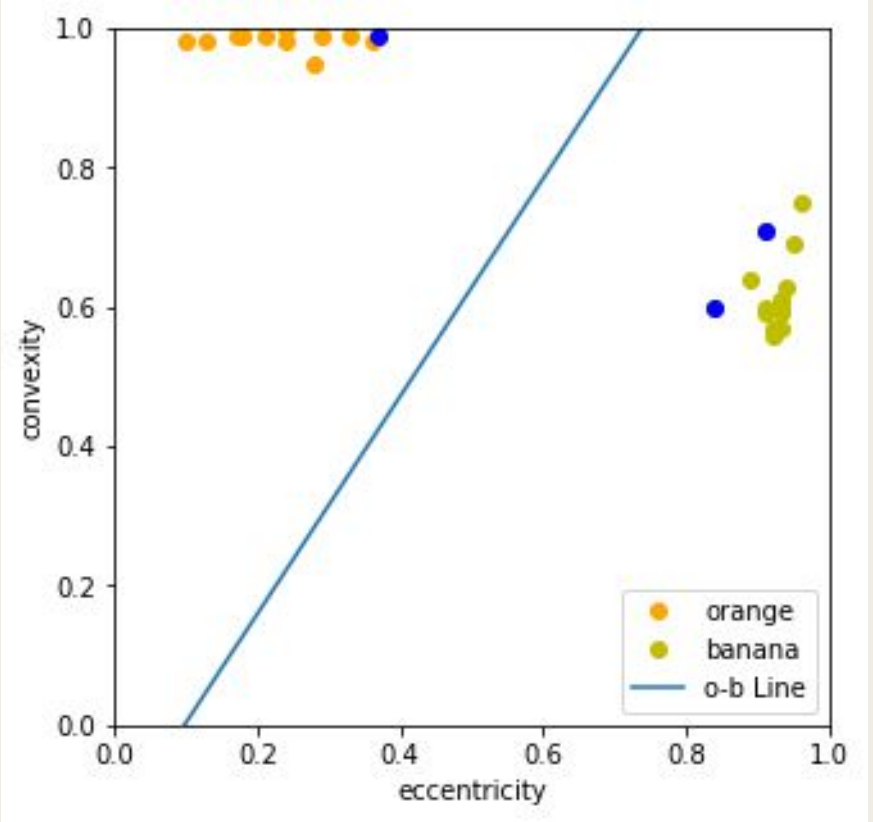
- I used qpsolvers' **solve_qp()** for the quadratic programming.
- When applying the quadratic programming a length N α vector was resulted. There are values that are not close to 0 thus those samples represent the **support vectors**.
- ***w*** is a m length vector that represents the weights per feature used.
- ***w*_0** can be solved using **x_1**, a support vector.

$$w = \sum_{i=1}^{n} \alpha_i z_i x_i = (\alpha .* z)^t x$$

$$w_0 = \frac{1}{z_1} - w^t x_1$$

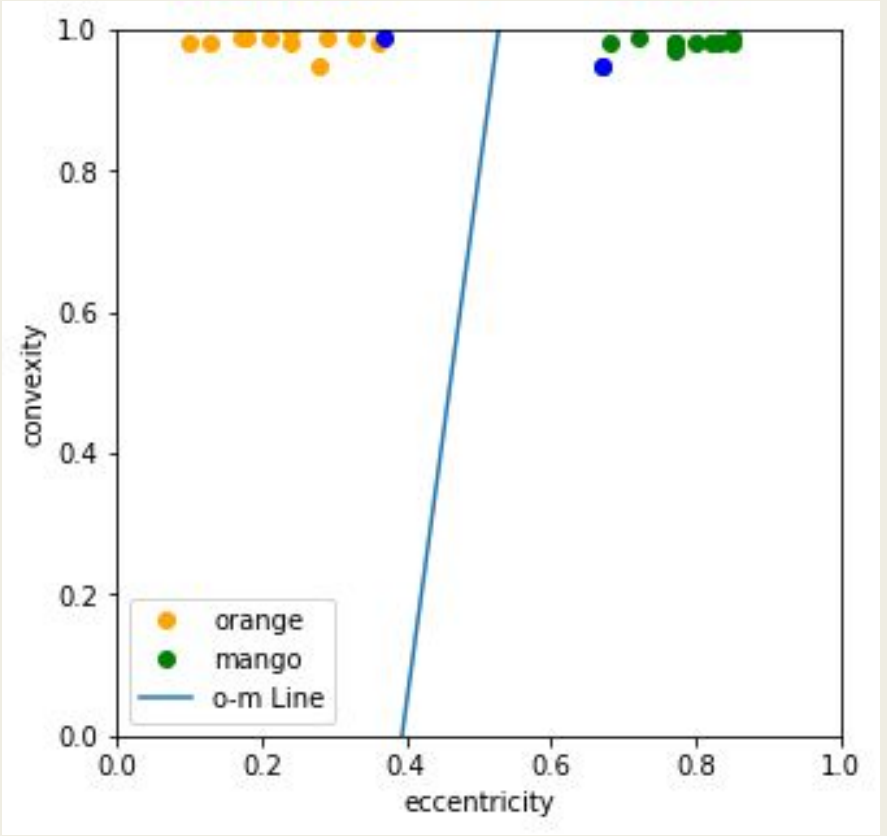
Results: Orange vs Banana

- Using the data of orange and banana, the algorithm found an optimized decision line using 3 support vectors (in blue).



Results: Orange vs Mango

- Using the data of orange and banana, the algorithm found an optimized decision line using 2 support vectors (in blue).



Results: Banana vs Mango

- Using the data of orange and banana, the algorithm found an optimized decision line using 2 support vectors (in blue).