My example is that of a person who wants to eat at least 2000 calories, but spend less than 60 minutes in the kitchen everyday. He has 3 options at his disposal:

- **option A**: cook a quick pasta dish
- **option B**: cook cheap dry beans, but it will take a long time
- **option C**: order fast-food, but it will be more expensive

He wants to minimize the money he spends on food while respecting the calorie and time constraints.

Below is the table detailing the price, calories and time needed to get one portion of food from each option.

	A	**B**	**C**
Calories	800	1600	400
Price	2	2	6
Time	25	1	5

Ex 4

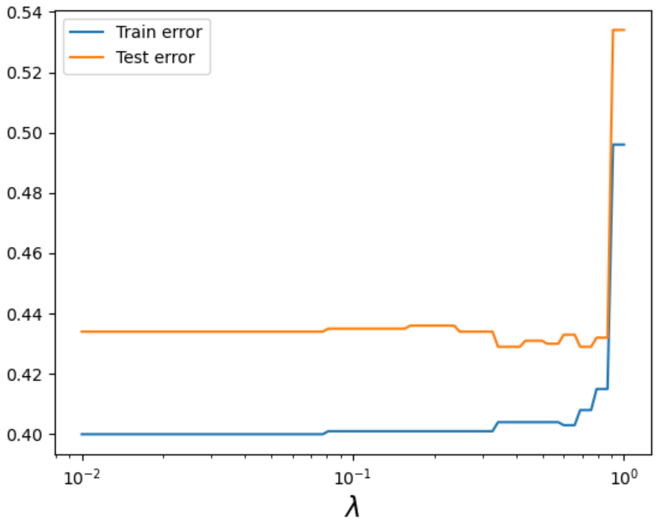
I chose to formulate a super vector machine classifier as an optimization problem.
The objective function that has to be minimized is:

$$f(\beta, \nu) = \frac{1}{m} * \sum_{i=1}^{m} (1 - y_i(\beta^T x_i - \nu)) + \lambda ||\beta||$$

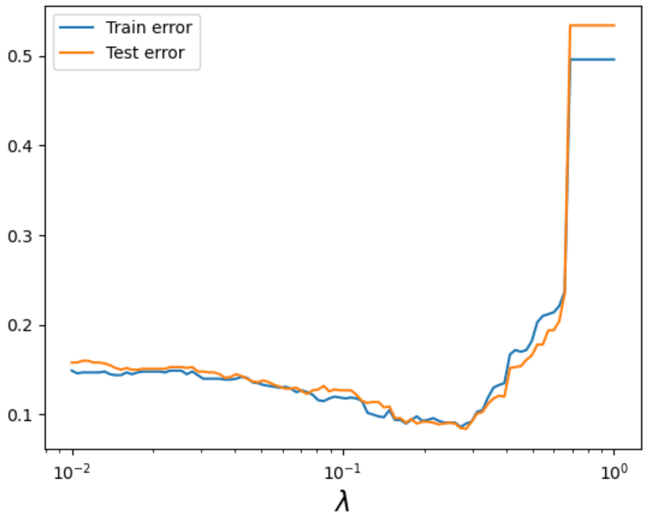
with xi being feature vectors, y1 the label set and $$\lambda$$ a regularization parameter.

I used both the l1 norm and the infinite norm to minimize this function without constraints, then with constraints in order to not use norm() and max() in the objective function.

The l1-norm equivalent with constraints gave a smoother trade off curve



than the l1 norm curve without constraints, using cvxpy.norm():



On the other hand, the infinity norm using max() and using constraints both produced almost identical trade off curves, suggesting the constraint solution is a good equivalent to get rid of max() in the objective function.