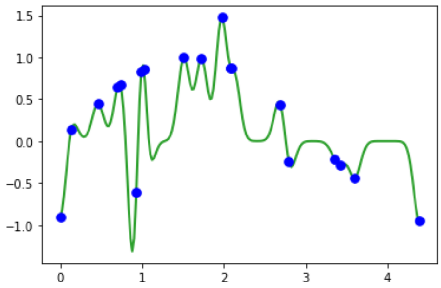
1 Kernel Ridge Regression

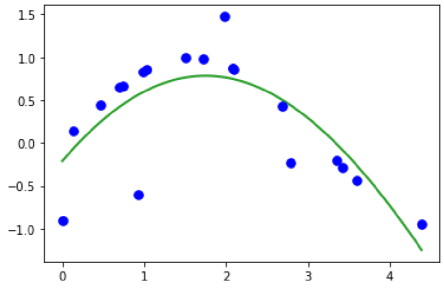
1.1 Conceptual Recap

1. Is kernel regression a parametric or nonparametric model? Explain.

   It is a nonparametric model. The kernel is applied to each data point, so the number of parameters increase as the number of data points increase.

2. Consider the RBF kernel, $k(x, x^{(i)}) = e^{-\gamma \|x - x^{(i)}\|_2^2}$. Match the models below with their corresponding γ values (0.01 or 100). What is the effect of γ on overfitting?





   The left model corresponds to $\gamma = 100$, while the right corresponds to $\gamma = 0.01$. When γ grows too large, the model is likely to overfit.

1.2 Kernelize it!

Recall the process for kernel regression:

Step 1: Compute $\alpha = (K + \lambda\mathbb{I})^{-1}y$ where: $K_{ij} = k(x^{(i)}, x^{(j)})$ and **k** is your kernel.

Step 2: Given a new point **x**, predict $\hat{y} = \sum_{i=1}^{N} \alpha_i k(x, x^{(i)})$
For this question, consider the box kernel:

$$k(x, x^{(i)}) = \begin{cases} 1 & \|x - x^{(i)}\|_2^2 \leq \frac{1}{2} \\ 0 & otherwise \end{cases}$$

You are given the following 1D data points (x,y): (3,4), (3.7,1), (4.2,-2). **a)a)**

3. **Find K and calculate α. For simplicity, let $\lambda = 0$. You can use an online inverse matrix calculator.**

   K (3x3): $\begin{bmatrix} 1 & 1 & 0 \\ 1 & 1 & 1 \\ 0 & 1 & 1 \end{bmatrix}$**, where $\mathbf{K}^{-1}$ (3x3):** $[r]01-1$
   $1-11$
   -110

   y (3x1): [4, 1, -2]

   With $\gamma = 0$, we get α (3x1): $\mathbf{K}^{-1}\mathbf{y}$ = $[r]01-1$
   $1-11$
   -110 **[4, 1, -2] = [3, 1, -3]**

4. **Predict $\hat{y}$ for x = 3.4.**

   $\hat{y} = \sum_{i=1}^{3} \alpha_i k(3.4, x^{(i)})$

   $= \alpha_1 k(3.4, 3) + \alpha_2 k(3.4, 3.7) + \alpha_3 k(3.4, 4.2)$

   = 3*1 + 1*1 + (-3)*0

   = 4

2 A Quick Intro to Constrained Optimization

The following example is taken from http://people.brunel.ac.uk/ mastjjb/jeb/or/morelp.html

Suppose a company makes two products (xylophones and yo-yos) using two machines (A and B). Each xylophones that is produced requires 50 minutes processing time on machine A and 30 minutes processing time on machine B. Each yo-yo that is produced requires 24 minutes processing time on machine A and 33 minutes processing time on machine B.

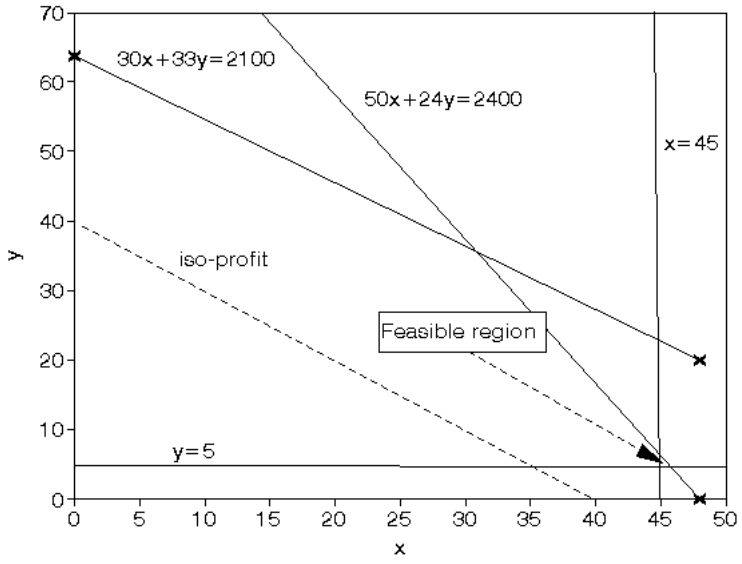
At the start of the current week there are 30 xylophones and 90 yo-yos stock. Available processing time on machine A is forecast to be 40 hours and on machine B is forecast to be 35 hours.

The demand for xylophones in the current week is forecast to be 75 units and for yo-yos is forecast to be 95 units. Company policy is to maximise the combined sum of the number of xylophones and yo-yos at the end of the week.

Formulate the problem of deciding how much of each product to make in the current week.

Let x be the number of xylophones produced in the current week, and let y be the number of xylophones produced in the current week.

We find the constraints:
$50x + 24y \leq 40 \cdot 60$ machine A time
$30x + 33y \leq 35 \cdot 60$ machine B time
$x \geq 75 - 30$ (so production of xylophones is at least demand minus initial stock)
$y \geq 95 - 90$ (so production of yo-yos is at least demand minus initial stock)
The objective is: $\max(x + 30 - 75) + (y + 90 - 95) = x + y - 50$ (maximizes the number of units left in stock at the end of the week)



Formulate the previous problem using vectors and matrices:

$\max x + y - 50$
s.t. $\begin{pmatrix} 50 & 24 \\ 30 & 33 \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} \leq \begin{pmatrix} 2400 \\ 2100 \end{pmatrix}, \begin{pmatrix} x \\ y \end{pmatrix} \geq \begin{pmatrix} 45 \\ 5 \end{pmatrix}$

More generally, we can say this problem has the following form:
$\max \mathbf{c}^T\mathbf{x}$
s.t. $A\mathbf{x} \leq \mathbf{b}, \mathbf{x} \geq \mathbf{0}$ (note that we can drop the -50 from the objective)

So now we have a concise, generalized way of expressing optimization problems. Refer to page 13 of the slides from October 28 to see the SVM optimization problem statement.

3 Duals

Motivating idea: for every maximization problem satisfying some certain set of criteria, there exists a corresponding minimization problem with the same optimal solution.
Using the above formulation, we write the dual as:
$\min \mathbf{b}^T\mathbf{y}$
s.t. $A^T\mathbf{y} \geq \mathbf{c}, \mathbf{y} \geq 0$

We can think of these problems are quasi-inverses/transposes of each other.

Some important theorems:
Let (P) be a maximization problem and (D) its corresponding dual.
Weak Duality: any feasible solution to (P) has value less than or equal to the value of any feasible solution to (D).
Strong duality: If the objective to (P) is convex, (P) has linear constraints, and (P) is feasible, then the optimal solutions to (P) and (D) have the same value.
Slater's condition: if (P) has a point that is strictly feasible, then strong duality holds (Slater point).

We introduced these formulations with the objectives being linear combinations of the variables, but in practice this need not always be true.
Note also that when strong duality holds, we have that $\mathbf{x}$ and $\mathbf{y}$ are respectively primal and dual feasible (this is because they are valid solutions to their own problems).

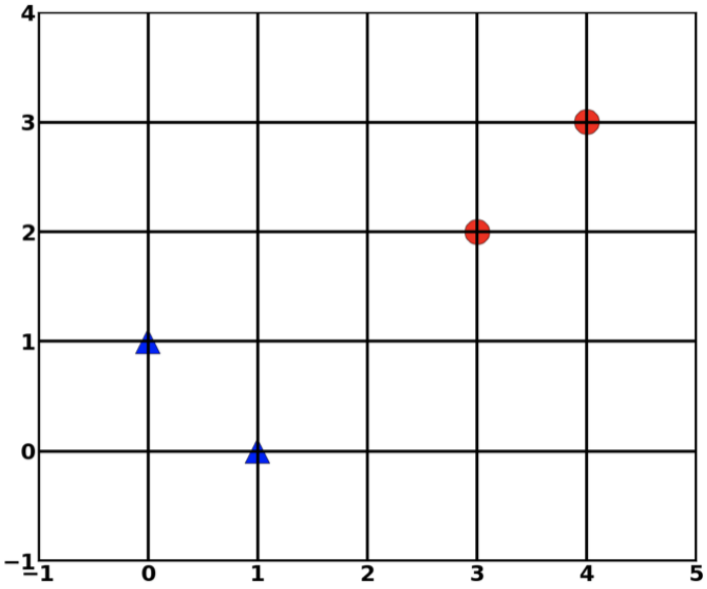
Complementary Slackness: At the optimal solution, we have that for every value x_i in $\mathbf{x}$, either $x_i = 0$ or the constraint corresponding to x_i in (D) is tight (active). Similarly, we also have that for every value y_j in $\mathbf{y}$, either $y_j = 0$ or the constraint corresponding to y_j in (P) is tight (active).
Why is this true?

Suppose $\mathbf{x}$ and $\mathbf{y}$ are optimal for (P) and (D). Then we know $\mathbf{c}^T\mathbf{x} = \mathbf{b}^T\mathbf{y}$
But by weak duality we know $\mathbf{c}^T x \leq \mathbf{y}^T A\mathbf{x} \leq \mathbf{y}^T\mathbf{b}$
So $\mathbf{b}^T\mathbf{y} = \mathbf{y}^T A\mathbf{x}$
Note that $\mathbf{b}^T\mathbf{y}$ is a scalar, so we can write $\mathbf{b}^T\mathbf{y} = \mathbf{y}^T\mathbf{b}$
$\Rightarrow \mathbf{y}^T(A\mathbf{x} - \mathbf{b}) = 0.$
Reading off the individual rows, we see this gives us the complementary slackness conditions.

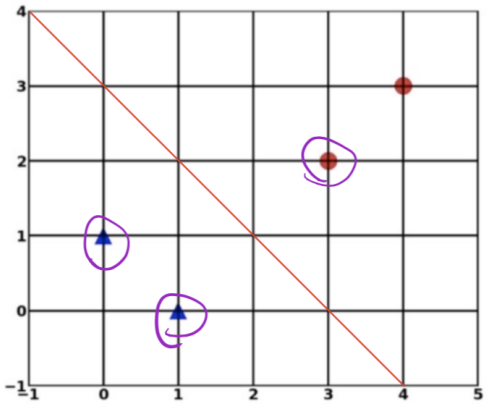
4 Support Vector Machines

Assume we are given dataset $\mathcal{D} = \{(\mathbf{x}_i, y_i)\}_{i=1}^n$, where $x_i \in \mathbb{R}^d$ and $y_i \in \{+1, -1\}$.
In SVM the goal is to find some hyperplane which separates the positive from the negative examples, such that the margin (the minimum distance from the decision boundary to the training points) is maximized. Let the equation for the hyperplane be $\mathbf{w}^T\mathbf{x} + b = 0$.

1. You are presented with the following set of data (triangle = +1, circle = −1):



Find the equation of the hyperplane $\mathbf{w}^T\mathbf{x} + b = 0$ that would be used by an SVM classifier. Which points are support vectors?



Support vectors are circled.

2. Let's try to measure the width of the SVM slab (we assume that it was fitted to linearly separable data). We can do this by measuring the distance from one of the support vectors, say $\mathbf{x}_+$, to the plane $\mathbf{w}^T\mathbf{x} + b = 0$ Since the equation of the plane is $\mathbf{w}^T\mathbf{x} + b = 0$ and since $c\mathbf{w}^T\mathbf{x} + b = 0$ defines the same plane, we have the freedom to choose the normalization of w. Let us choose normalization such that

$\mathbf{w}^T\mathbf{x}_+ + b = +1$ and $\mathbf{w}^T\mathbf{x}_- + b = -1$, for the positive and negative support vectors respectively. Show that the width of an SVM slab with linearly separable data is $\frac{2}{||\mathbf{w}||}$.

Let d_+ denote the margin from $\mathbf{x}_+$ to the plane $\mathbf{w}^T\mathbf{x} + b = 0$. Let d_- denote the margin from $\mathbf{x}_-$ to the plane. The width then can be written as:

$$\begin{aligned}
width &= d_+ + d_- \\
&= \frac{\mathbf{w}^T\mathbf{x}_+}{||\mathbf{w}||} + b - \frac{\mathbf{w}^T\mathbf{x}_-}{||\mathbf{w}||} - b && \text{(Project } x \text{ to } w\text{, normalize and translate)} \\
&= \frac{\mathbf{w}^T\mathbf{x}_+}{||\mathbf{w}||} - \frac{\mathbf{w}^T\mathbf{x}_-}{||\mathbf{w}||} \\
&= \frac{1-b}{||\mathbf{w}||} - \frac{-1-b}{||\mathbf{w}||} && \text{(Since both } x_+ \text{ and } x_- \text{ are support vectors)} \\
&= \frac{1}{||\mathbf{w}||}((1-b) - (-1-b)) \\
&= \frac{2}{||\mathbf{w}||}
\end{aligned}$$

Thus we've proved that the width is $\frac{2}{||\mathbf{w}||}$

3. Write SVM as an optimization problem. Conclude that maximizing the margin is equivalent to minimizing $||\mathbf{w}||$. In other words, write the SVM as some $max_{w,b}\frac{2}{||\mathbf{w}||}$ with certain constraints and show that the maximization problem implies $min_{w,b}||\mathbf{w}||$ on some constraints.

   The main idea for SVM is to find a linear separator with a maximum margin. Using what we've proved in b), we can write SVM as follows:

$$\begin{aligned}
&max_{w,b}\frac{2}{||\mathbf{w}||} \\
s.t. &y^{(i)}(\mathbf{w}^T\mathbf{x}^{(i)} + b) \geq 1 \qquad \forall i
\end{aligned}$$

   By inverting the objective function, we get the following equivalent optimization problem:

$$\begin{aligned}
&min_{w,b}\frac{||\mathbf{w}||}{2} \\
s.t. &y^{(i)}(\mathbf{w}^T\mathbf{x}^{(i)} + b) \geq 1 \qquad \forall i
\end{aligned}$$

   Note that in the constraint the $\mathbf{x}^{(i)}s$ are not only the support vectors, but every data point.

4. Will moving points which are not support vectors further away from the decision boundary effect the SVM's the w and b returned by the SVM optimization?

   No. Since SVM only compute w and b based on support vectors, which are positive and negative samples that lie on the margin.

5 LaGrange Multipliers

Solve the following constrained optimization problem:

$$\min_{x,y} f(x,y) = x^2 + y^2$$

Subject to

$$g(x,y) = 0$$

where

$$g(x,y) = x + y - 1$$

We use the method of LaGrange Multipliers and explain some intuition for why it works at the end.

We introduce the LaGrangian $L(x,y,\lambda) = f(x,y) - \lambda g(x,y)$

The gradient is then $\nabla L(x,y,\lambda) = \nabla f(x,y) - \nabla \lambda g(x,y)$

We then set equal to 0 and solve the system:

$$\frac{\partial L}{\partial x} = 0$$
$$\frac{\partial L}{\partial y} = 0$$
$$\frac{\partial L}{\partial \lambda} = 0$$

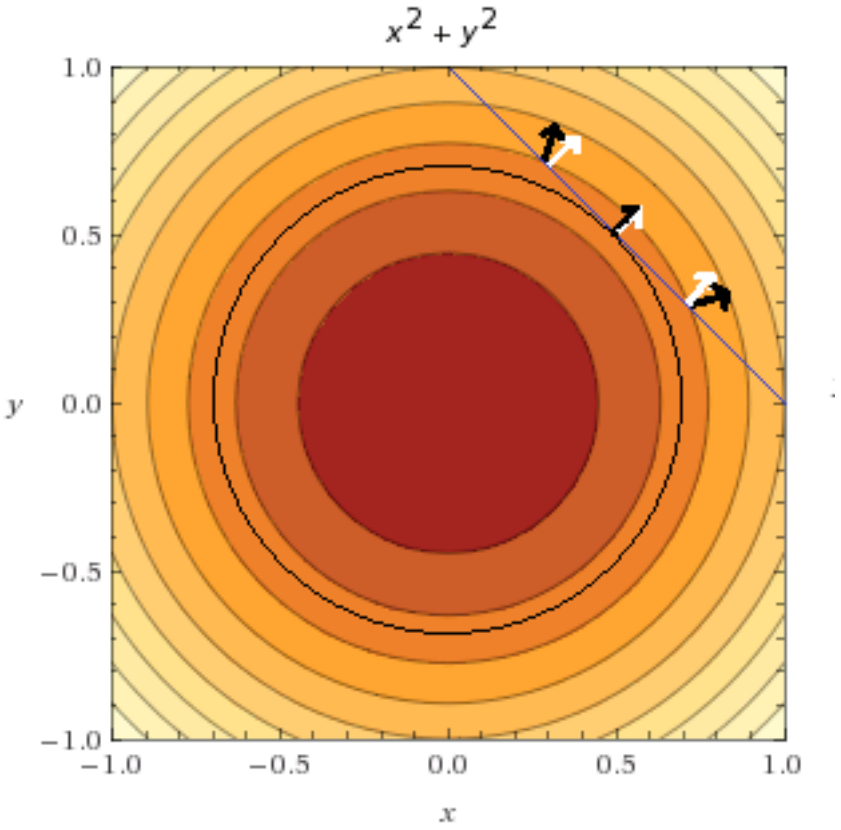
This then give us the system:

$$2x - \lambda = 0$$
$$2y - \lambda = 0$$
$$x + y = 1$$

Solving the resulting system gives us the optimum at $x = \frac{1}{2}, y = \frac{1}{2}$

5.1 Geometric Intuition for LaGrange Multipliers

Consider the previous problem. The main observation for why LaGrange multipliers work is to see that the minimum of $f(x, y)$ under the constraint $g(x, y)$ occurs when their gradients are parallel.
Geometrically, we have the following contour plot:



The circles are the contours of the objective function $f(x, y) = x^2 + y^2$ with darker colors signalling smaller values. The line in the top right is the constraint $g(x, y) = y = 1 - x$. Black arrows signal the gradient of $f(x, y)$ and white arrows signal the gradient of $g(x, y)$. The claim is that the middle point, where the gradients are parallel (in the same direction in this case), is the minimum of the objective under the constraint.

To see this, suppose you start at the leftmost point (on the line labelled with the 3 points and gradients) Say that the objective function evaluated at this point gives us some value v.

You would like to go left, but then you see that the gradient of the objective function is pointing towards the left (remember it is always pointing towards greater values) so it might not be a good idea. To be sure, you try and go to the left. You find out that the objective function returns some value $x > v$, so it is increasing which is not okay. Going to the right, this time, the objective function returns smaller values, it is good you are going in the correct direction. So you continue like that until you reach the center point where the two arrows are parallel. But then, once you move a little bit to the right you find out that the objective function is increasing again. As you can not move right or left anymore without increasing the objective function, you conclude this point is the minimum.

Finally, how does λ come into play? We know that the minimum occurs where the gradients are in the same direction. But their magnitudes may be different. Thus, λ is the factor where $\nabla f(x, y) = \lambda \nabla g(x, y)$ and $g(x, y) = 0$.

For a more in-depth look please see https://www.svm-tutorial.com/2016/09/duality-lagrange-multipliers. Credit is given to this website as well for the problem and explanation.

We have now seen the four KKT conditions for strong duality:

1. Solution to the primal is feasible.
2. Solution to the dual is feasible.
3. Solutions satisfy complementary slackness.
4. Gradient of the LaGrangian at optimal solutions is zero.

These conditions are good to know since they apply for non-linear optimization problems as well, so they give us a nice way to check for optimality. Specifically, since SVM is an instance of Quadratic Programming (QP), we can use them to verify optimality to a solution for SVM.

6 Return to SVM

We have the primal formulation:
$\min_{\mathbf{w},b} \frac{1}{2}\mathbf{w} \cdot \mathbf{w}$ such that $\forall\, j, (w \cdot x + b)y_j \geq 1$

This time, we'll set up the Lagrangian dual. To do this, attempt to solve the primal problem using Lagrange multipliers. To formulate this as the dual, interpret each Lagrange multiplier as a dual variable, and maximize the resulting expression over the dual variables such that each dual variable is non-negative.

Using the Lagrangian, we formulate the dual:
$\max_{\alpha} \min_{\mathbf{w},b} L(\mathbf{w}, b, \alpha)$
$= \max_{\alpha} \min_{\mathbf{w},b} \frac{1}{2}\mathbf{w} \cdot \mathbf{w} - \sum_j \alpha_j[(\mathbf{w} \cdot \mathbf{x}_j + b)y_j - 1], \alpha_j \geq 0$

Kernel Trick

Suppose we have feature vectors $x = \begin{pmatrix} x_1 & x_2 & x_3 \end{pmatrix}^T$ and $y = \begin{pmatrix} y_1 & y_2 & y_3 \end{pmatrix}^T$. If we were using an SVM, we would want to (1) broadcast these into a higher dimensional space and (2) take the resulting dot product.

Suppose wanted to broadcast the vectors to polynomials of degree 2. What would the resulting x and y vectors look like? How many steps would be involved in computing $x \cdot y$?

extend x to $\begin{pmatrix} x_1 & x_2 & x_3 & x_1x_2 & x_2x_3 & x_1x_3 & x_1^2 & x_2^2 & x_3^2 \end{pmatrix}$ and likewise we do the same to y.
To compute $x \cdot y$ would require we compute 9 products and take their sum.

However, notice that $x \cdot y = (1 + x_1y_1 + x_1y_2 + x_3y_3)^2$. This computation requires four products and one sum. While in one example this isn't a huge speedup, in practice this can be extremely valuable.

Recall the formulation for SVM:

$\max_\alpha \sum_i \alpha_i - \frac{1}{2} \sum_{i,j} \alpha_i \alpha_j y_i y_j \mathbf{x}_i \cdot \mathbf{x}_j$ such that $\sum_i \alpha_i y_i = 0, \alpha_i \geq 0$
How many dot products do we do each time we compute the objective value?

We see that we do $i \times j$ many dot products every time we compute the objective value. Clearly, for a large data set, we will need a fast way to compute the dot product $\mathbf{x}_i \cdot \mathbf{x}_j$.

Now suppose we have a data set with 10 features, and we wish to use polynomial features of degree 3. How many features are in the new data set? How many multiplication operations does it take to compute the dot product of two data vectors?

1,000 features in the broadcast data set. To compute a single dot product requires 1,000 multiplications.

Now suppose we use the kernel trick. How many multiplications do we do now?

With the kernel trick, computing the product only takes 12 multiplications ($(x_1y_1 + ... + x_{10}y_{10})^3$, one for each value and one for the cubing). Using the kernel trick, we've reduced our multiplications from 1,000 to 12.

More generally, if we have n features and we wish to use polynomial degree p, we have n^p features in the broadcast data set. Computing the dot product requires we add the product of every pair of features, meaning we need to do out n^p multiplications. However, using the kernel trick reduces this to $n + (p - 1)$ computations. So using the kernel trick changes this computation from being $O(n^p)$ to being $O(n)$.

Interesting consequence: we never write out the new, higher dimensional features anywhere.
Why SVMs work; Occam's Razor.