This notebook contains an excerpt from the [Python Data Science Handbook](https://jakevdp.github.io/PythonDataScienceHandbook/) by Jake VanderPlas; the content is available [on GitHub](https://github.com/jakevdp/PythonDataScienceHandbook).

Machine Learning from Data - IDC 2022

Assignment 5: ID: 203074224, 208930503

Question 1

Let $K(x,y)$=(x·y+1)3 be a function over ℝ2×ℝ2 (i.e., x,y∈ℝ2).

Find ψ for which $K$is a kernel. (It may help to first expand the above term on the right-hand side).

$$K(x, y) = (\overrightarrow{X} * \overrightarrow{Y} + 1)^3$$

$$(x_1 y_1 + x_2 y_2 + 1)^3$$

$$Note : (a + b + c)^3 = a^3 + b^3 + c^3 + 3a^2 b + 3a^2 c + 3b^2 a + 3b^2 c + 3c^2 a + 3c^2 b + 6abc.$$

$$(x_1 y_1)^3 + (x_2 y_2)^3 + 1 + 3(x_1 y_1)^2 x_2 y_2 + 3(x_1 y_1)^2 + 3(x_1 y_1)(x_2 y_2)^2 + 3(x_2 y_2)^2 + 3x_1 y_1 + 3x_2 y_2 + 6x_1 y_1 x_2 y_2$$

$$x_1^3 y_1^3 + x_2^3 y_2^3 + 1 + 3x_1^2 y_1^2 x_2 y_2 + 3x_1^2 y_1^2 + 3x_1 y_1 x_2^2 y_2^2 + 3x_2^2 y_2^2 + 3x_1 y_1 + 3x_2 y_2 + 6x_1 y_1 x_2 y_2$$

Equals to a vectorial product of:

$$< (x_1^3, x_2^3, 1, \sqrt{3}x_1^2 x_2, \sqrt{3}x_1^2, \sqrt{3}x_1 x_2^2, \sqrt{3}x_2^2, \sqrt{3}x_1, \sqrt{3}x_2, \sqrt{6}x_1 x_2), (y_1^3, y_2^3, 1, \sqrt{3}y_1^2 y_2, \sqrt{3}y_1^2, \sqrt{3}y_1 y_2^2, \sqrt{3}y_2^2, \sqrt{3}y_1, \sqrt{3}y_2, \sqrt{6}y_1 y_2) >$$

We define:

$$\phi(\overrightarrow{x}) = (x_1^3, x_2^3, 1, \sqrt{3}x_1^2 x_2, \sqrt{3}x_1^2, \sqrt{3}x_1 x_2^2, \sqrt{3}x_2^2, \sqrt{3}x_1, \sqrt{3}x_2, \sqrt{6}x_1 x_2)$$

So:

$$K(x, y) = \phi(\overrightarrow{x}) * \phi(\overrightarrow{y})$$

b. What did we call the function ψ in class if we remove all coefficients?

Lets look at $\phi(x)$ without the coefficiets: $\phi(\overrightarrow{x}) = (x_1^3, x_2^3, 1, x_1^2 x_2, x_1^2, x_1 x_2^2, x_2^2, x_1, x_2, x_1 x_2)$

We multiply each feature in $\overrightarrow{X}$ in "every combination" untill we get to the power of 3 -> a.k.a Full Rational Varaity, Cubic because we get the power of 3. (Lecture 7, Linear Claiffier in higher dimentions, page 27)

c. How many multiplication operations do we save by using $K(x,y)$ versus $\psi(x)\cdot\ \psi(y)$?

In the Kernal: $K(x, y) = (\overrightarrow{X} * \overrightarrow{Y} + 1)^3 = (x_1 y_1 + x_2 y_2 + 1)^3$ we have 4 multiplications.

In the mapping procedure, when we also multiply the coefficients (and with considering sqrt of coefficient as constant):

$\phi(\overrightarrow{x}) = (x_1^3, x_2^3, 1, \sqrt{3}x_1^2 x_2, \sqrt{3}x_1^2, \sqrt{3}x_1 x_2^2, \sqrt{3}x_2^2, \sqrt{3}x_1, \sqrt{3}x_2, \sqrt{6}x_1 x_2)$

mapping has 2, 2, 0, 3, 2, 3, 2, 1, 1, 2 operations = 18 multiplications for each ϕ

The multiplication of $\phi(\overrightarrow{x}) * \phi(\overrightarrow{y})$ takes 10 operations So in total we have: 18*2 + 10 = 46 operations.

We save 42 operations (91% better performance, nice)

Question 2

Let $f(x,y)=2x-y$. Find the minimum and the maximum points for f under the constraint $g(x,y)=\frac{x^2}{4}+y^2=1$.

define :

$$f(x,y,\lambda)=2x-y+\lambda(x^2/4+y^2=1)$$
$$\Delta f(x,y,\lambda)=(2+\lambda\frac{x}{2},-1+\lambda 2y,\frac{x^2}{4}+y^2-1)$$

$$(2+\lambda\frac{x}{2},-1+\lambda 2y,\frac{x^2}{4}+y^2-1)=(0,0,0)$$

meaning:

$$2+\lambda\frac{x}{2}=0$$
$$-1+\lambda 2y=0$$
$$\frac{x^2}{4}+y^2-1=0$$

meaning:

$$\lambda x=-4$$
$$\lambda y=\frac{1}{2}$$
$$x^2+4y^2=4$$

Assign 1,2 equations to the third:

$$(\frac{-4}{\lambda})^2+4(\frac{1}{2\lambda})^2=4$$

$$\frac{16}{\lambda^2}+\frac{4}{4\lambda^2}=4$$

$$\frac{16}{\lambda^2}+\frac{1}{\lambda^2}=4$$

$$\frac{17}{4}=\lambda^2$$

$$\pm\frac{\sqrt{17}}{2}=\lambda$$

Assign lambda to x, y equations:

$$\pm\frac{\sqrt{17}}{2}x=-4$$
$$\pm\frac{\sqrt{17}}{2}y=\frac{1}{2}$$

meaning:

$$\pm\sqrt{17}x=-8$$
$$\pm\sqrt{17}y=1$$

meaning:

$$x=\mp\frac{8}{\sqrt{17}}$$

corrolating with

$$y=\pm\frac{1}{\sqrt{17}}$$

Meaning these are extreme points by Lagrange theorem under the necessary constrains:

$$(x_1,y_1)=(-\frac{8}{\sqrt{17}},\frac{1}{\sqrt{17}})$$
$$(x_2,y_2)=(\frac{8}{\sqrt{17}},-\frac{1}{\sqrt{17}})$$

Easy trick, we just assign x and y to the original function to classify min\max:

$$f(x_1, y_1) = -\frac{18}{\sqrt{17}}$$
$$f(x_2, y_2) = \frac{17}{\sqrt{17}}$$

first is the min, latter is the max point :)

Question 3:

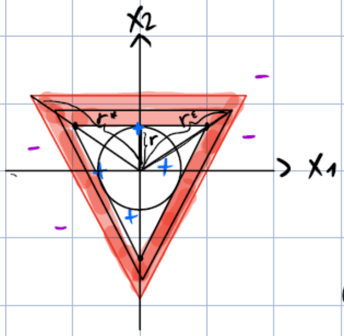
Let $X=\mathbb{R}2$. Let vectors $u=(\sqrt{3}/2,1/2)$, $w=(\sqrt{3}/2,-1/2)$, $v=(0,-1)$ and $C=H=\{h(r)= \{(x1,x2)|(x,y)\cdot u\le r,(x,y)\cdot v\le r(x,y)\cdot w\le r,\}\}$, for $r>0$.

The set of all origin-centered upright equilateral triangles.

Describe a polynomial sample complexity algorithm L that learns C using H. State the time complexity and the sample complexity of your suggested algorithm. Prove all your steps.

3) נתון: $C=H=\left\{h(r)=\left\{(x_1,x_2) \,\middle|\, \begin{matrix}(x,y)\cdot u\le r,\\ (x,y)\cdot v\le r,\\ (x,y)\cdot w\le r\end{matrix}\right\}\right\}, r>0$

פיתרון:



נגדיר $r=0$.

יהי r^* המגדיר את משולש הקונספט $C(r^*)\in C$ שאליו אנחנו שואפים להתקרב.

לכל $d\in D$ כש $D\subseteq R^2$ וגם $d=(x_1,x_2)$

וגם $x_1,x_2\in R$ נחשב:

$$R=\max\{d\cdot u, d\cdot v, d\cdot w\}$$

אם $R>r \Leftarrow r=R$

לבסוף נחזיר את $L(D)=h(r)\in H$

<u>time complexity</u>: לעבור על כל המופעים ייקח לנו $O(m)$.

לחשב max לכל מופע ייקח לנו $O(1)$

לעדכן r כמידת הצורך ייקח לנו $O(1)$.

סה"כ $O(m)$.

<u>sample complexity</u>:

יהי ϵ אחוז הטעות המקסימלי. נרצה שיתקיים $Pr[(x_1,x_2)\in A_r]\le\epsilon$

כאשר A_r הוא הפרש השטחים בין משולש הקונספט לבין $h(r)$ שהחזרנו

← הטבעת המשולשית הצבועה באדום.

נגדיר:

$$r^{\epsilon} = \arg\inf_r \Pr[(x_1,x_2)\in A_r] \le \epsilon$$

כלומר הטבעת (המלבנית) הגדולה ביותר עם סיכוי של לכל היותר ϵ.

נפצל את ההוכחה לשני מקרים:

<u>מקרה 1</u>: נניח $r^{\epsilon} \le r$ אזי בוודאות ההסתברות לטעות/להימצא בתוך הטבעת קטן מ ϵ.

<u>מקרה 2</u>: נניח $r > r^{\epsilon}$. נראה שיתקיים שהסיכוי שנטעה ביותר מ ϵ מוגבל על δ. כלומר,

$$\Pr[\text{error}_D(h(r)) \ge \epsilon] \le \delta.$$

ההסתברות שכל m דגימות אקראיות כך שלא יפלו בטבעת הינו $(1-\epsilon)^m \le e^{-\epsilon m}$.

עבור סך דגימות כגודל $m \ge \frac{\ln(\frac{1}{\delta})}{\epsilon}$ נקבל:

$$e^{-\epsilon m} \le e^{-\ln(\frac{1}{\delta})} = e^{\ln(\delta)} = \delta$$

* אם הסיכוי ליפול בטבעת הוא נורא קטן, למרות הפרש השטחים שנוצר בין r ל r^*, עדיין גודל הטעות יהיה קטן מאוד מהתוצאה.

סה"כ מצאנו שסיבוכיות הדגימות והזמן הן פולינומיאליות.

Question 4:

A business manager at your ecommerce company asked you to make a model to predict whether a user is going to proceed to checkout or abandon their cart. You created the model using, and reported 20% error on your test set of size 1000 samples. In the business manager's presentation to upper management, he presented your model and stated that the company can expect 20% error when deploying the model live on the website. Luckily, you realize that this is a mistaken assumption, and you correct the statement to say that **with 95% confidence**, the **true error** they can expect is up to what percentage? (Just state the error percentage).

In [1]:
```
n = 1000.0
p = 0.20
se = (p*(1-p)/n)**0.5
print(f"We can expect for {(p-2*se)*100}% up to {(p+2*se)*100}% true error")
```

```
We can expect for 17.470177871865296% up to 22.529822128134704% true error
```

HW5 - Question 5 - SVMs

Instructions

The objective of this exercise is to allow you to experiment with how different slack coefficients (refered to as `C` values in this notebook) affect the SVM model, and in particular, the train and test accuracy.

The provided code below generates data, fits an SVM model, plots the SVMs with the margin, and plots the accuracies as a function of `C`.

Currently, the code runs for only 2 values of `C`. Your task is to add at least 5 values to the `Cs` array indicated below, in order to achieve the desired accuracy graph that appears at the bottom of the notebook.
Your graph does not need to be identical, but should present similar behavior as appears in the desired graph.

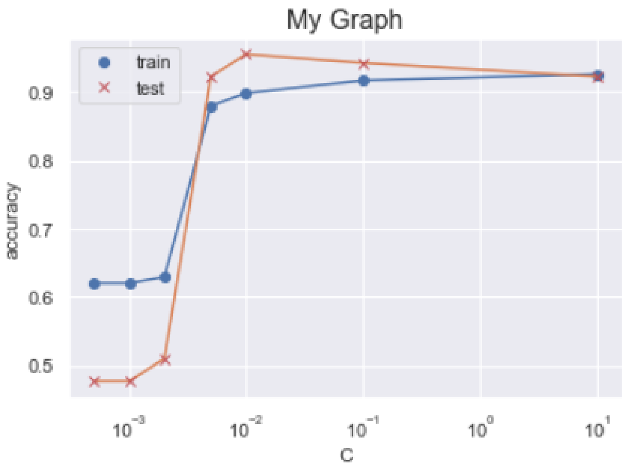
When you are finished, take a screenshot of your achieved graph and paste it into your submission PDF along with your responses to the theoretical questions. DO NOT submit your code.

Array of `C` values

Add at least 5 values to the list in the cell below.

In [5]:
```
# add at least 5 values to this list to achieve the desired accuracy graph at the bottom of the notebook

Cs = [0.0005, 0.001, 0.002, 0.005, 0.010, 0.1, 10]
```



Desired accuracy graph

Below is the graph you want to immitate.

[8]:
```
from IPython.display import Image
Image(filename='desired_graph.png', width=400, height = 2(
```

Out[8]: