# 7 Problems and Solutions (Part 1 of 3)

## Kernel Regression

Background: Consider applying linear regression to ____________data. We could use a polynomial like $y = w_1 x_i + w_2 x_i^2 + b$ and check graphically (or by comparing $\text{MSE}_{\text{train}}$ and $\text{MSE}_{\text{test}}$) for a good fit. However, for $D > 3$ dimensions, finding the right polynomial could be hard.

*Kernel regression* is a non-parametric method. It extends weighted $k$-NN (another non-parametric method) to the case of ____________

- Its simplest form for 1D $\mathbf{x} = x$ is $f(x) = \sum_{i=1}^{N} w_i y_i$, where $w_i = \frac{k\left(\frac{x - x_i}{b}\right)}{\sum_{j=1}^{N} k\left(\frac{x - x_j}{b}\right)}$.[1]
- $f(x)$ is a ________________________ of $\{y_i\}$ since $\sum w_i = 1$.
- $k()$ is a *kernel* that plays the role of a similarity function. Coefficients $w_i$ are higher when $x$ is similar to $x_i$ and lower otherwise.
- The most common kernel is the ____________ kernel, $k(z) = \frac{1}{\sqrt{2\pi}} \exp\left(\frac{-z^2}{2}\right)$.

  The *bandwidth* $b$ is a hyperparameter tuned using cross-validation.

  e.g. Draw $\{(x_i, y_i)\}$ in nonlinear pattern. Add bell curves (wide, narrow, just right).

## Python[2]

```
from localreg import localreg, rbf
```

`y_hat = localreg(x, y, x0=None, degree=0, kernel=rbf.gaussian, radius=1)`
computes $\hat{y} = f(x)$ where

- `x` and `y` are values to fit
- `x0` are values at which to compute $\{\hat{y}_i\}$; the default is to use `x`
- `kernel` is a function of one argument (there are many kernels besides `rbf.gaussian`; we can make our own)
- `degree=0` corresponds to our weighted average
- `radius` is $b$

To learn more: `https://pypi.org/project/localreg/`

---

[1] For vector input, $x_i - x$ is replaced by Euclidean distance $||\mathbf{x}_i - \mathbf{x}||$. Burkov uses factors $\frac{1}{N}$ in front of his $f(x)$ sum and $N$ in his $w_i$ definition; I omitted them since $N\frac{1}{N} = 1$. Burkov uses $x_i - x$ and $x_j - x$ where I used their opposites to see that we're standardizing $x$ using $\mu = x_i$ and $\sigma = b$; $k()$ is symmetric, so this matters only for clarity (and compatibility with his §9).

[2] I have not found kernel regression in `skikit-learn`. However, the package `localreg` does it.