Machine Learning for Renewable Energy Forecasting

Yan Zhang

Professor, University of Oslo, Norway

Learning Objectives

Throughout this lecture, it is aimed for the students to be able to

- Understand forecasting uncertainty and why we need to forecast renewable energy production
- Learn basic machine learning techniques in plain language, programming language, and mathematical language
- Apply different machine learning techniques for wind energy forecasting

Industry Invited Talk Today

- **Speakers**: **Arne Gravdahl,** ***CTO & Founder, WindSim AS; Associate Professor, Norwegian University of Life Sciences***

- **Title**: **Wind Energy Forecasting: industry perspective**

- **WindSim**: **pioneered the use of CFD (*computational fluid dynamics)* technology to optimize wind turbine placement. More information: http://www.windsim.com/**

Outline

- **Wind Energy in Norway**
- **Wind Energy Forecasting**
  - **Wind power & wind speed curve**
  - **Forecasting definition**
- **Machine Learning Techniques and Their Applications in Wind Energy Forecasting**
  - **Linear Regression (LR)**
  - **K-Nearest Neighbor (kNN)**
  - **Support Vector Regression (SVR)**

Renewable Energy Forecasting

- Wind energy
  - Depend on wind and weather in general
  - Main focus in this lecture

- Solar energy
  - Depend on sun, cloud, and weather in general

- Wave energy
  - Wave energy converter are floating on the ocean surface waves
  - Wave power is the energy from ocean surface waves. The capacity depends on wave

Offshore & Onshore

Wind turbines can be installed:

- **Onshore: on land**
  - **Cheaper installation**
  - **Cheaper integration**
  - **Cheaper maintenance**
- **Offshore: on sea**
  - **Less obstruction**
  - **Higher and more steady wind speed**

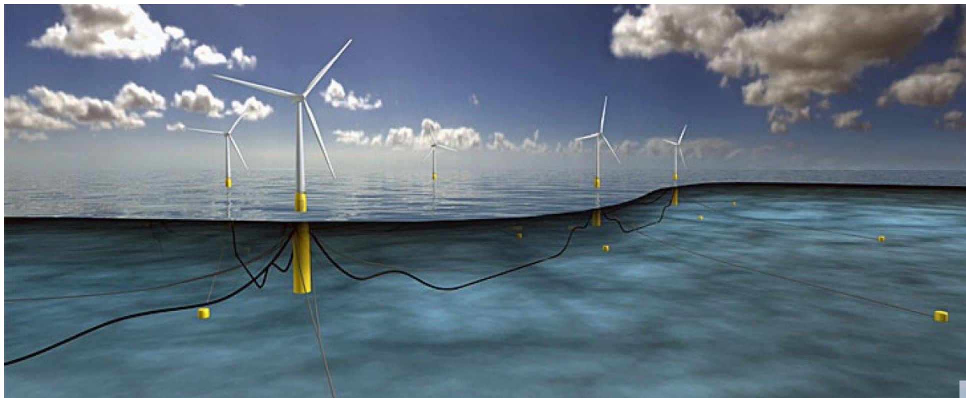
Q: what is the advantage of offshore wind farm?

Onshore wind farm

Offshore wind farm

You may watch 2-minute video at: http://www.youtube.com/watch?v=tsZITSeQFR0

Equinor (original name: Statoil) has great interest in offshore wind power



Floating wind turbines at Hywind SCOTLAND

ARKONA, Germany

https://www.equinor.com/en/what-we-do/new-energy-solutions/our-offshore-wind-projects.html

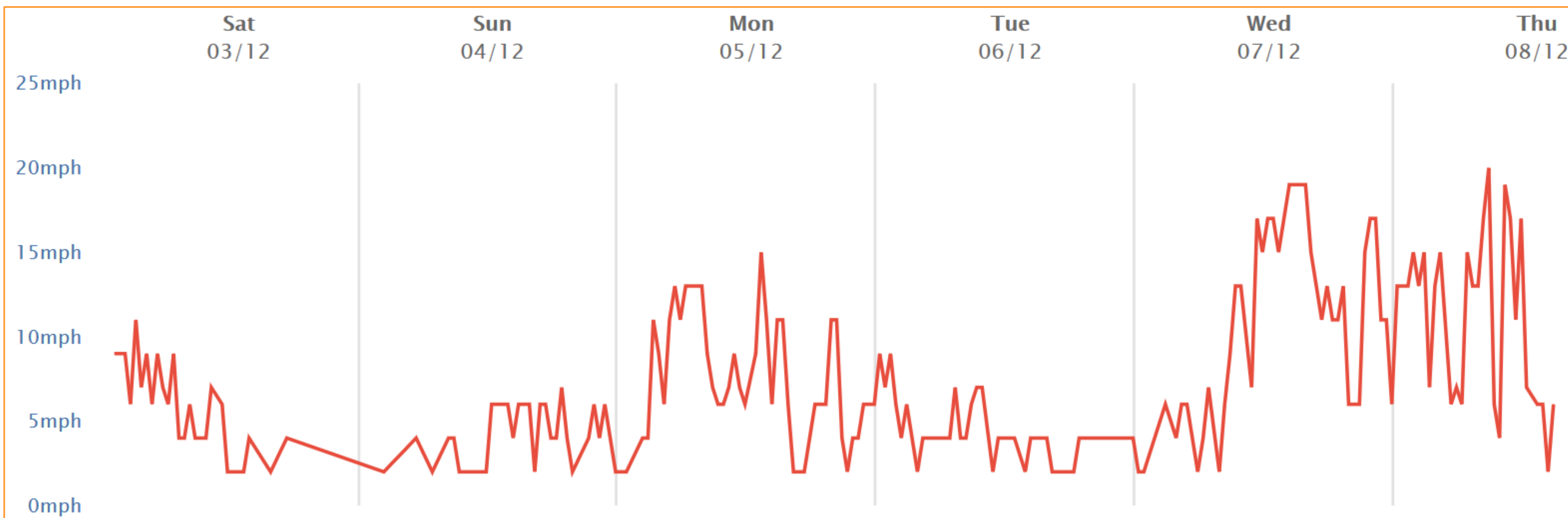
Wind forecasting and wind energy forecasting are important

- *Energy cost of intermittent wind:* unforecasted spin-regulation waste 2.5-7.5% of total energy
- *System reliability:* unforecasted ramp events may compromise reliability
- *Economic value:* good forecasts may lead to high economic value
- *Effective power grid management:* forecasts are essential for effective grid management with high wind penetrations (>5%)

A £2million, 100metre-tall wind turbine caught fire in hurricane force wind at Ardrossan, North Ayrshire, Scotland (2011). http://www.dailymail.co.uk/news/article-2071633

Wind speed forecasting is challenging

- **The key problem is the intermittency. Changes in wind speed will result in changes in wind power.**



Wind speed in Gardermoen (Oslo Airport) during December 3-8, 2016

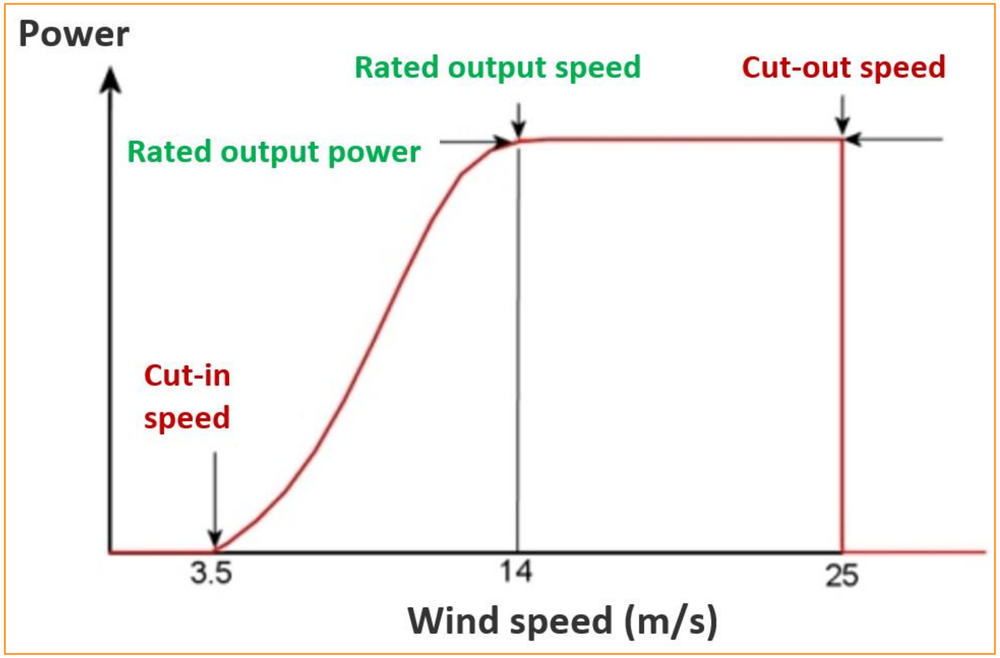
Data source: http://magicseaweed.com/Norway-Live-Winds/52/

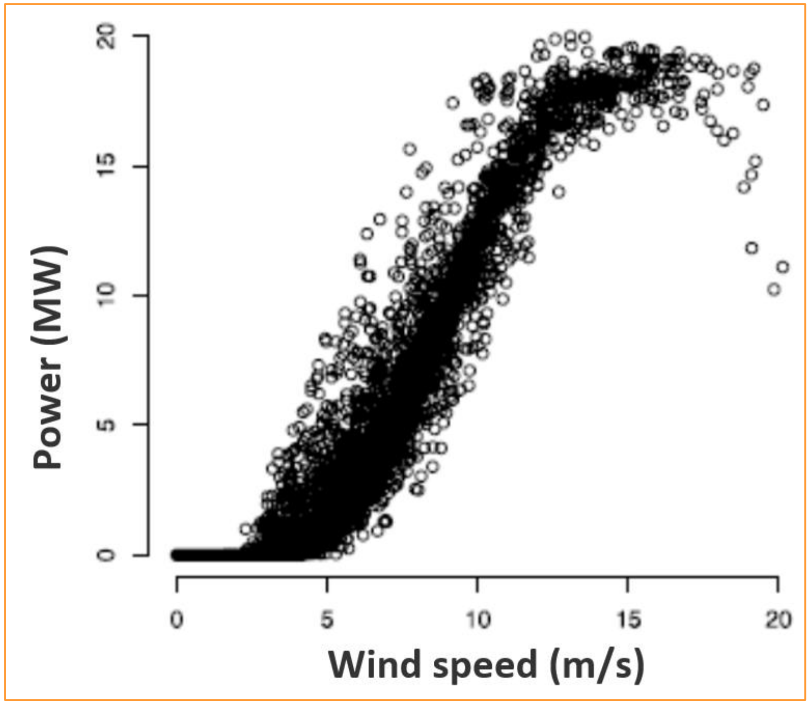
Wind energy forecasting is challenging

- Small pressure gradients over large distances: hard to forecast accurately
- Turbulent & chaotic processes are important: even harder to forecast
- Local topography can have strong influence: not in standard weather models
- Wind-power curves are highly nonlinear: small errors in wind = big errors in power
- Abnormal plants activities: malfunctions, downtime and sub-optimality

FORECASTING PROBLEM

Wind Power & Speed Curve

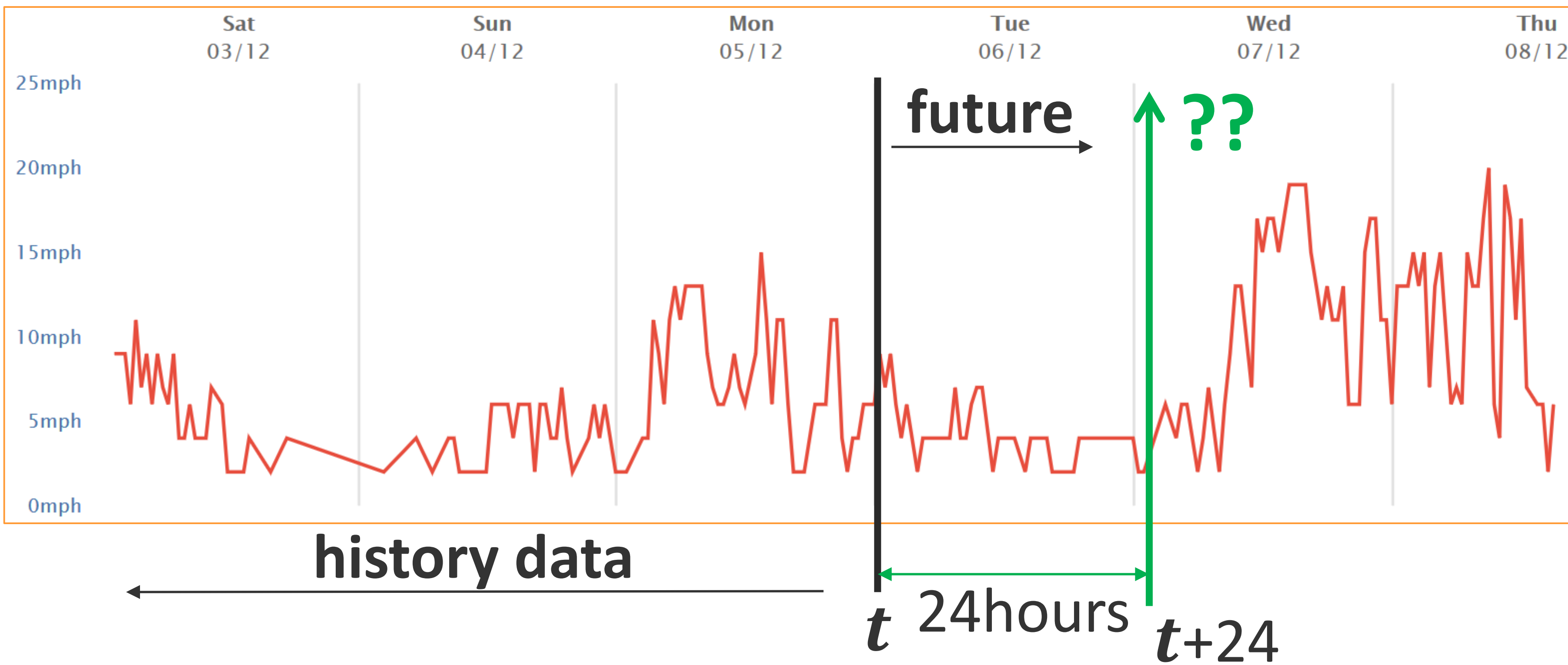




- Wind power depends on wind speed. Predicting wind speed can help us predict wind power.
- A minimum cut-in speed is needed to start power generation
- Denmark Klim wind farm: example empirical power curve for the wind farm over a 6-month period in 2002, based on hourly measurements of wind speed and corresponding power output.

Forecasting – wind speed in Oslo Airport as an example

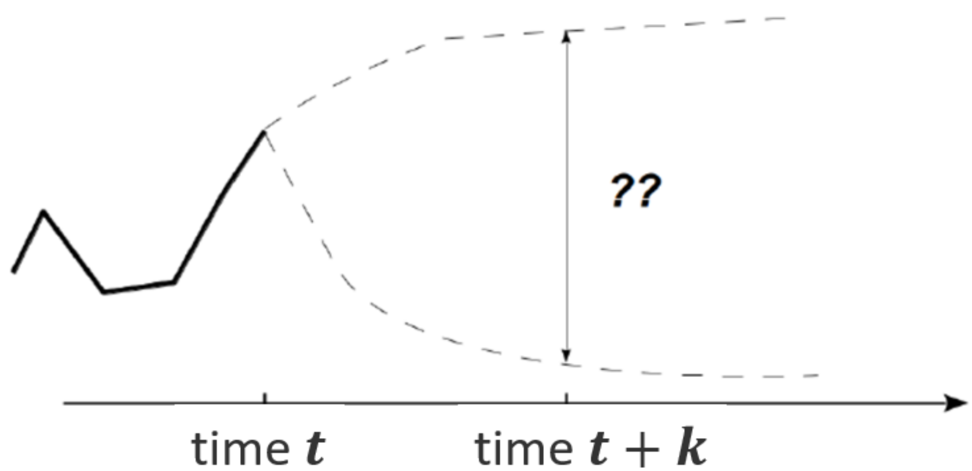
- A forecast: an estimate of wind speed at time t +24, conditional to all history data up to time t



Forecasting definition

The practical setup:

- **we are at time t (e.g., at 9am) and have all information up to time t**
- **and interested in what will happen at time $t + k$ (e.g., at 11am)**
- **k : referred to as the lead time**
- **Y_{t+k} : the random variable “power generation at time $t + k$”**

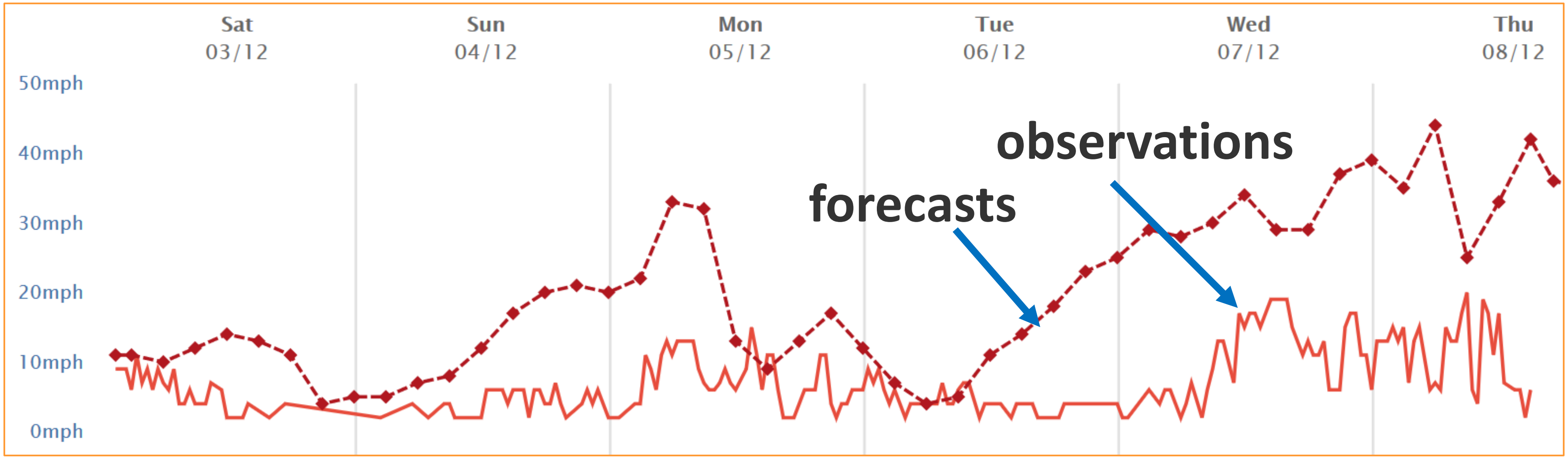


- **$\widehat{Y}_{t+k|t}$: the conditional expectation of power generation at time $t + k$, conditional to all information up to time t**

Forecasting definition – mathematical model

$$\hat{Y}_{t+k|t} = f(Y_{t+k}|Y_t, w_t, T, P, S)$$

- Y_t: history data up to time t
- Y_{t+k}: power generation at time t+k
- $\hat{Y}_{t+k|t}$: predicated value of power generation at time t+k
- w_t: wind speed at time t
- T: temperature
- P: pressure
- S: other parameters (e.g. wind direction)



Wind speed observations and forecasts in Gardeomen (Oslo Airport) during December 3-8, 2016

Different value of k for Time-scale classification for wind energy forecasting

Time-scale	Range	Applications
Very short-term	Few minute to 1 hour ahead	• Electricity market clearing • Real-time grid operations • Regulation actions
Short-term	1 hour to several hours ahead	• Economic load dispatch planning • Load reasonable decisions • Operational security in electricity market
Medium-term	Several hours to 1 week ahead	• Unit commitment decisions • Reserve requirement decisions • Generator online/offline decisions
Long-term	1 week to 1 year or more ahead	• Maintenance planning • Operation management • Optimal operating cost • Feasibility study for design of the wind farm

MACHINE LEARNING REGRESSION TECHNIQUES

What is machine learning?

- **Typical machine learning process**

New data:

Temperature: 10

Sunny: Yes

Go out for Beer: YES or NO?

training data

Machine learning approach

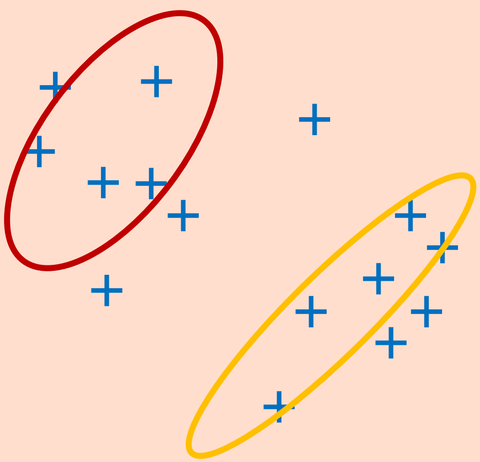
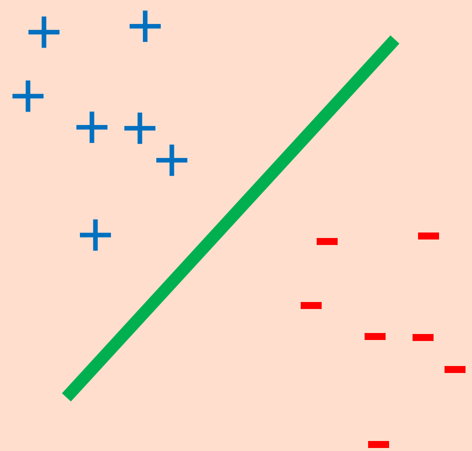
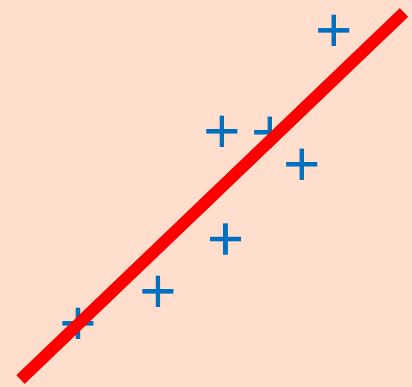
Estimation function, e.g., linear regression function

New prediction:

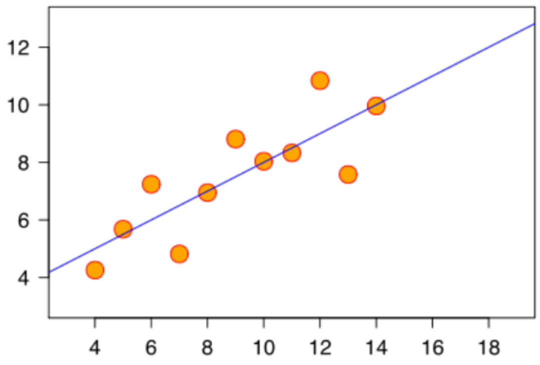
Go out for Beer: YES

Temperature	Sunny	Go out for Beer
20	Yes	YES
-5	No	NO
15	Yes	YES
2	Yes	NO

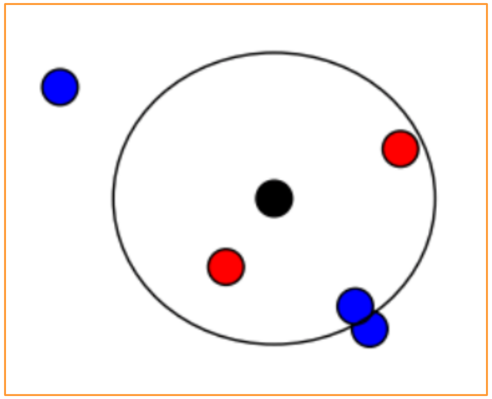
Machine learning overview

Clustering	Classification	Regression **(this lecture)**
		
K-means	• Decision tree • Linear Discriminant Analysis • Neural Networks • Support Vector Machines	• Linear regression • k Nearest Neighbors • Support Vector Regression
Group data based on their characteristics	Separate data based on their labels	Find a model that can explain the output given the input

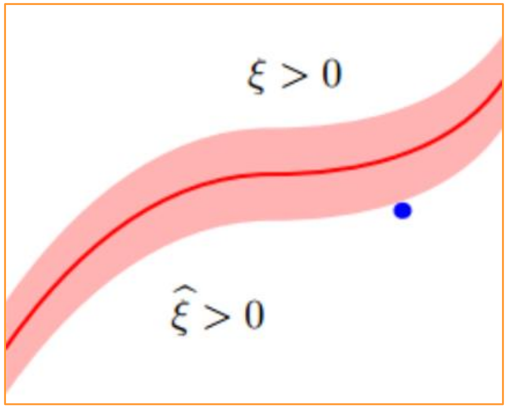
Three Regression Techniques



- **Linear Regression**

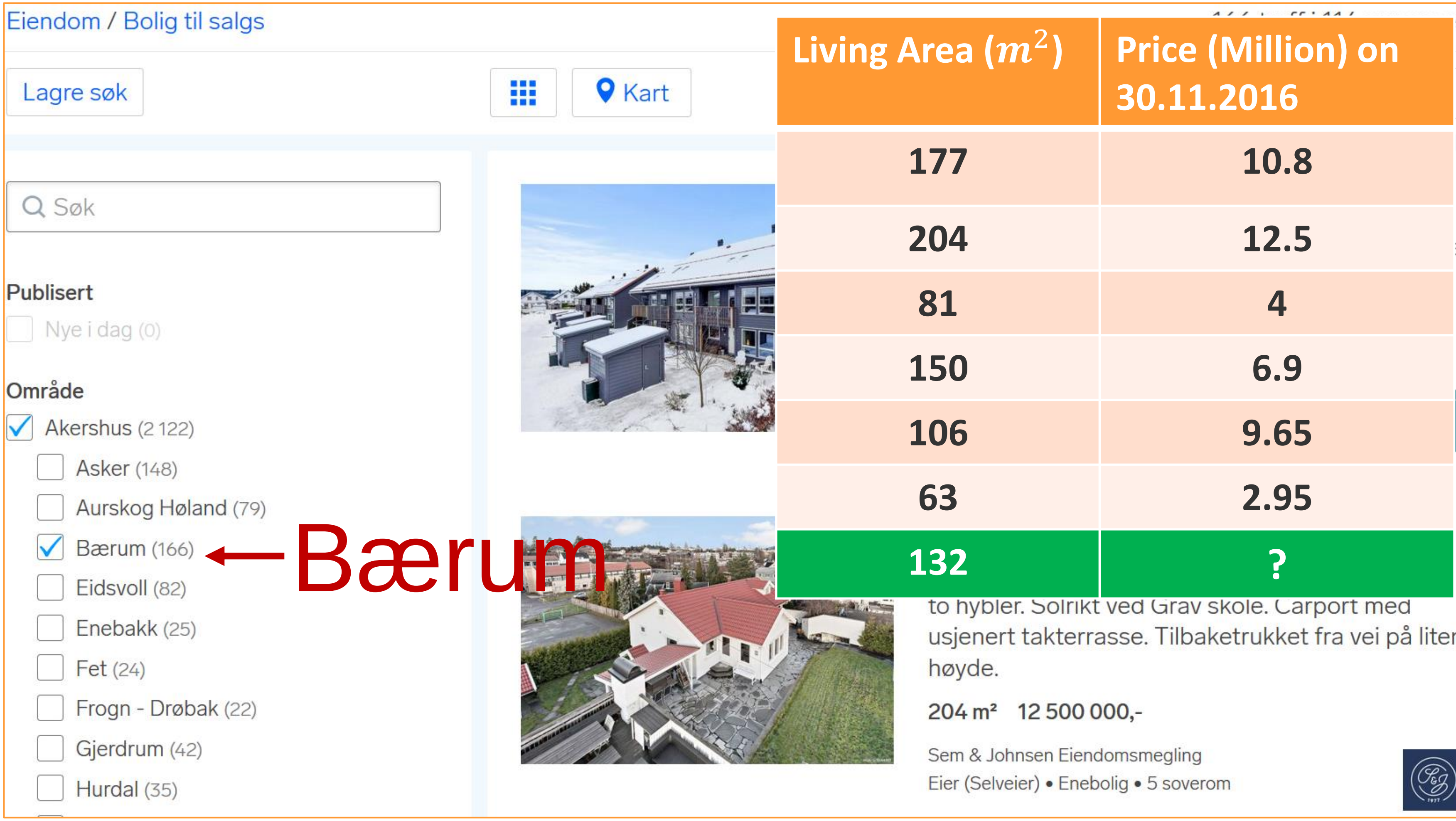


- **K-Nearest Neighbor Regression (kNN)**



- **Support Vector Regression (SVR)**

Linear Regression Model - real estate example



Living Area (m^2)	Price (Million) on 30.11.2016
177	10.8
204	12.5
81	4
150	6.9
106	9.65
63	2.95
132	?

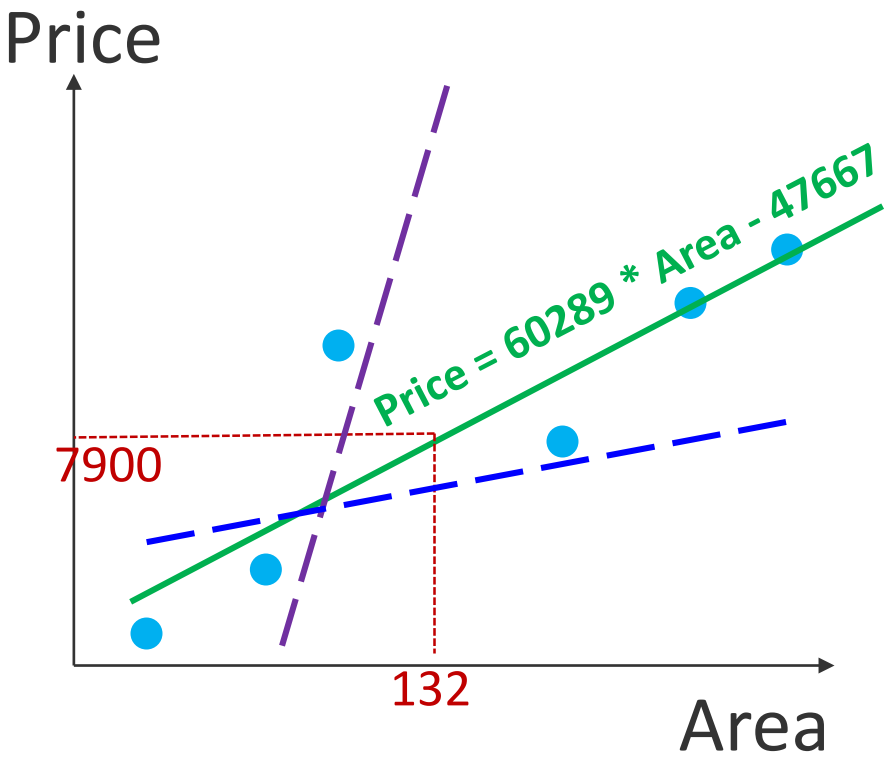
- **Q:** what is the price for a new house with area 132 m^2?

Linear regression model – main idea

- **6 points are plotted**
- **Find a straight line that models price as a function of living area. For example:**

  Price = 60289 * Area - 47667

- **Then, we can calculate the estimated price 7.9M for a house with area $\mathbf{132}m^2$**



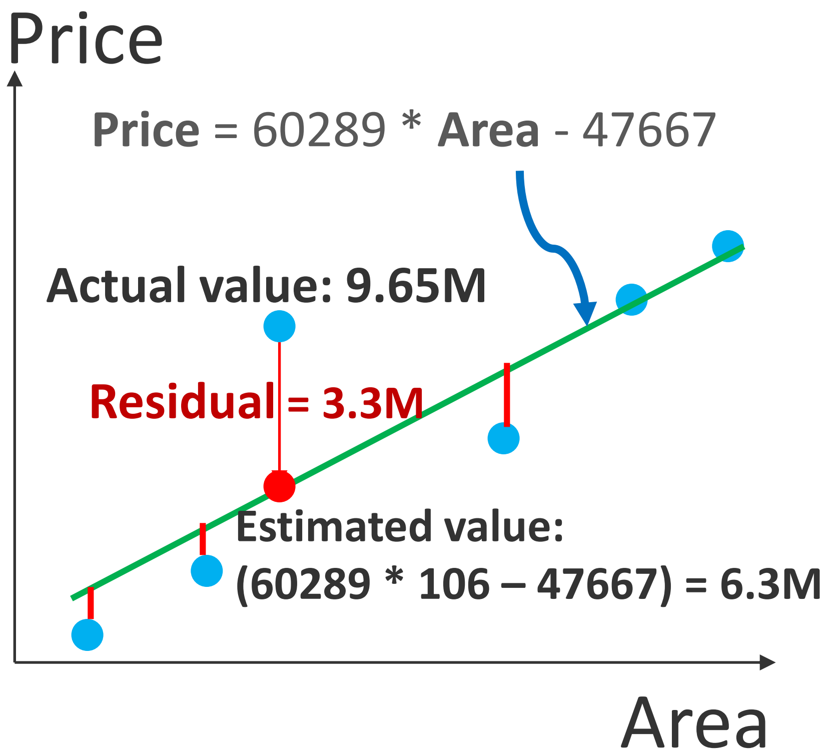
- **Q: how did you get the linear function "Price = 60289 * Area – 47667" ? Is it a good model? → we need the concept of residual**

Linear regression model – residual

- **Not all observations are perfectly on the line. There is error between the actual value and the estimated value by the linear model**

- ***A residual*: is defined as the difference between the actual value and the estimated value by the linear approximation.**

- **We have interest in minimizing the overall error between the linear model and the actual observations. Here, overall error refers to: the sum of residual squares.**



Linear regression model

- Given N observations $(\boldsymbol{x_i}, \boldsymbol{y_i})$; $(\mathbf{i} = \mathbf{1}, \mathbf{2}, \dots \mathbf{N})$.
- *Goal:* to find the linear function to minimize the overall error between the model and the actual observations.
- The linear model is defined as: $\boldsymbol{y} = w_0 + w_1 \boldsymbol{x}$
- The residual for observation $\boldsymbol{i}$ is given by

$$e_i = \underset{\text{Actual value}}{y_i} - \underbrace{(w_0 + w_1 x_i)}_{\text{Estimated value}}$$

- The overall error between the linear model and the actual observations is defined as the sum of residual squares. Hence, we need to find coefficients w_0 and w_1 that minimize

$$\min_{\boldsymbol{w}} \sum_{i=1}^{N} e_i^2$$

Least square is a standard approach to solve this problem

- **Least square**: a mathematical procedure for finding the best estimation function to a given set of points by minimizing the sum of the squares of the residuals of the points. Results have closed-form expressions

$$\begin{cases} w_1 = \sum_{i=1}^{N}(x_i - \bar{x})(y_i - \bar{y}) / \sum_{i=1}^{N}(x_i - \bar{x})^2 \\ w_0 = \bar{y} - w_1 \bar{x} \end{cases}$$

where $\bar{x} = \frac{\sum_{i=1}^{N} x_i}{N}$; $\bar{y} = \frac{\sum_{i=1}^{N} y_i}{N}$

- **Matlab polyfit function solves linear regression problem**

```
p = polyfit(x,y,n)
```

→ **x, y: two N-by-1 vectors of data value**

→ **p: coefficient of polynomial of degree n**

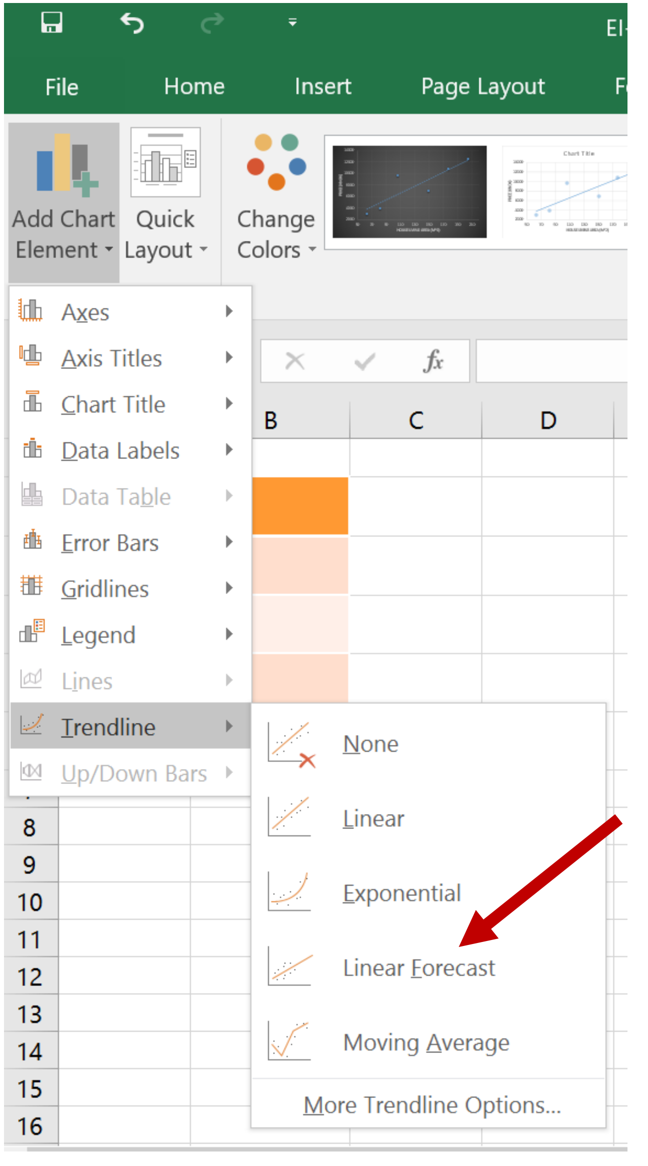
coeffecient = polyfit(x,y,1)

For more information: https://se.mathworks.com/help/matlab/ref/polyfit.html

Linear regression in Python or Excel

- **Scikit-learn is an open sources machine learning tool in Python. It is built on NumPy, SciPy, and matplotlib. More information: www.scikit-learn.org**

```
import matplotlib.pyplot as plt
import numpy as np
from sklearn import datasets, linear_model

# Create linear regression object regr =
linear_model.LinearRegression()

# Train the model using the training sets
regr.fit(X_train, y_train)

# The coefficients
print('Coefficients: \n', regr.coef_)
```



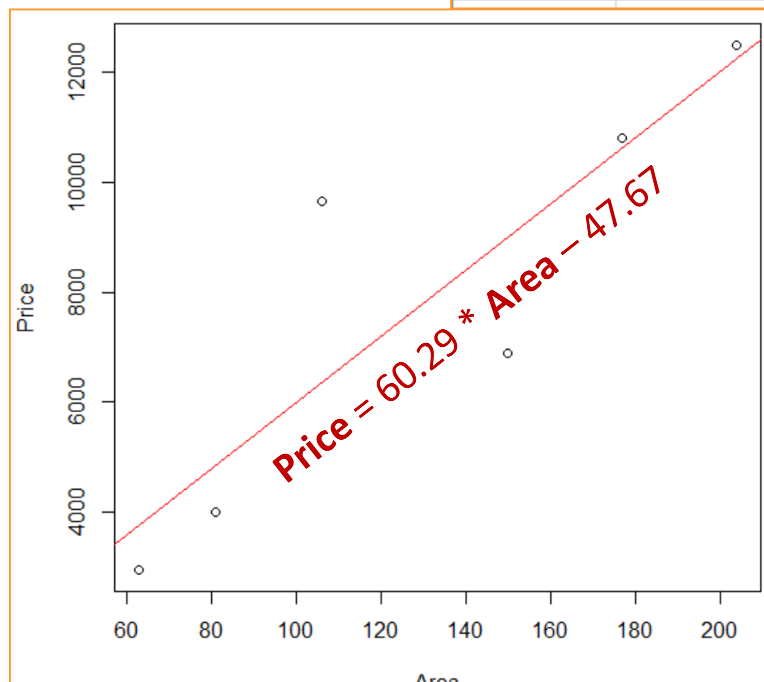
Linear regression in R

- R is a free software environment for statistical computing and graphics
- Installation and more information: https://www.r-project.org/

```
#read CSV file

MyData <- read.csv("MyData.csv", sep=";",
header=TRUE)

#scattplot points

plot(Price ~ Area, data = MyData)

#linear regression function

lm.out = lm(Price ~ Area, data = MyData)

#plotting the regression line on an existing
scatterplot

abline(lm.out, col='red')

lm.out
```

MyData.csv

Area	Price
177	10800
204	12500
81	4000
150	6900
106	9650
63	2950



Output:

```
Coefficients:
(Intercept)         Area
     -47.67        60.29
```

Linear Regression for Wind Energy Forecasting

	A	B	C	D	E	F	G
1	TIMESTAMP,POWER,U10,V10,WS10						
2	20120101 1:00,0.2736781568,0.5348940035,-3.6602432799,3.6991204986						
3	20120101 2:00,0.0867959455,0.3308130989,-2.6764297561,2.6967969048						
4	20120101 3:00,0.0068114015,-0.0658387244,-2.0290719396,2.0301398163						
5	20120101 4:00,0.0186459868,-0.4195494463,-1.7990895574,1.847361625						
6	20120101 5:00,0.0348118328,-0.7542244222,-1.6615260214,1.8246981117						

Normalized wind power

Wind speed above ground 10m

- The data is between 2012.01.01-2013.10.01, which has 15336 data records
- The file contains hourly wind power measurements and wind speed
- “WS10”: wind speed above ground 10m
- “POWER”: real wind power measurement (normalized) for a wind farm in Australia (Q: why normalized wind power data?)
- Normalized data to preserve anonymity

Training data and test data

- For all 15336 data records, we divide them into two parts.
- **Training data**: The first 15000 data records are used as training data to find the linear function (i.e., w_0 and w_1)
- **Test data**: The last 336 data records are used as test data. This shows the period (from 2013-09-17 01:00:00 to 2013-10-01 00:00:00). Remind that these data are true value of wind power and wind speed in such period. We can calculate 336 predicted wind power generation based on the linear regression model. Then we compare the difference between the predicted value and the true value to calculate the prediction accuracy.
- In general case, we may use 80% data of all data records as the training data while the last 20% data as the test data.

Flowchart of machine learning

- We have 336 test data record (wind power, wind speed).
- Now, we enter 336 wind speed data to the linear model

Input 15000 training data

Linear regression approach lm(.)

Linear function:
Power=0.12333 * Speed – 0.08603

- Calculate 336 predicted wind power generation
- Compare the predicted value and the true value

Find the Training Model, i.e., linear function

- **The wind speed in the 336 test data records are used in the linear function to find the predicted wind power generation.**
- **The true wind power in the test data are then used to compare with the forecasted power generation. This can show if the training model (i.e., linear function) can provide accurate prediction.**

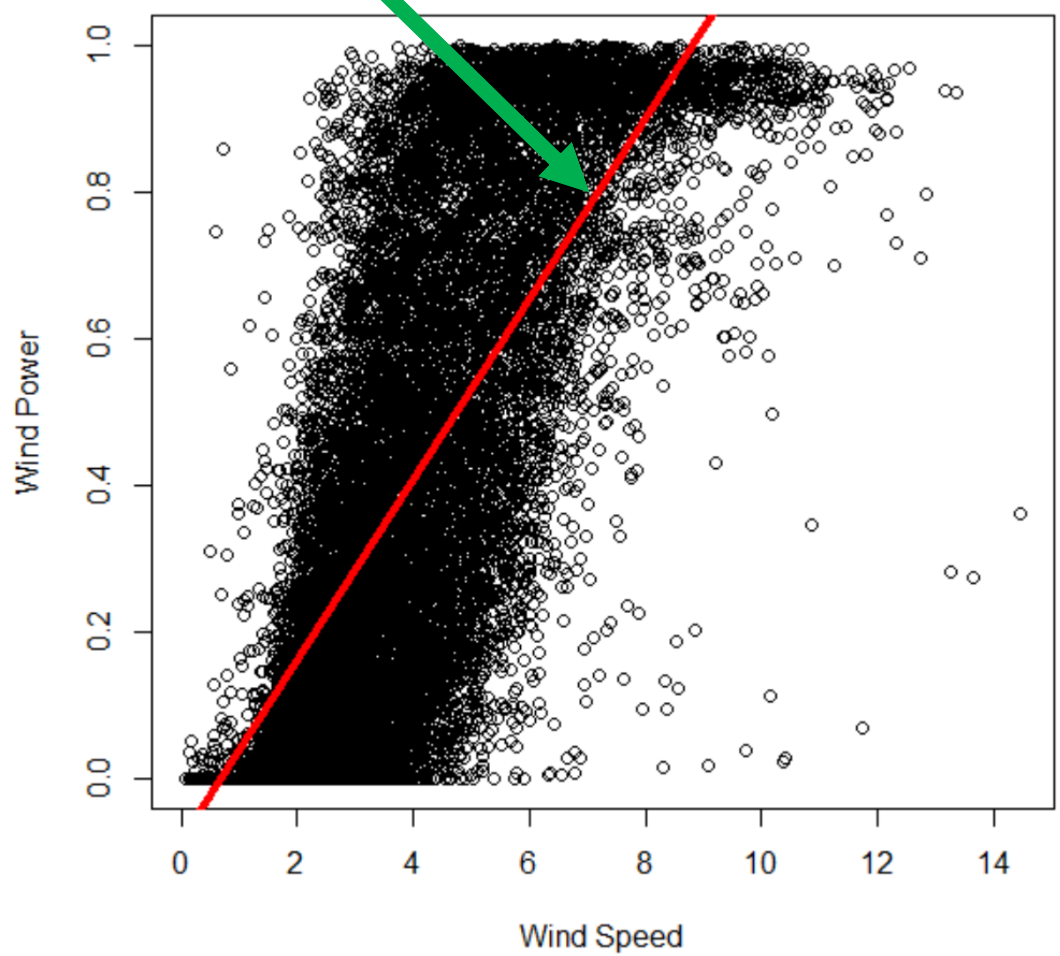
```
#read data from CSV file
data <- read.csv("WindPowerData.csv)

#the first 15000 data are training data
powerTrain = data$POWER[1:15000]
wsTrain = data$WS10[1:15000]

#linear regression
lmOut = lm(powerTrain ~ wsTrain)
#get the coefficients in the linear model and save in coeffs
coeffs = coefficients(lmOut)
```

linear regression

Linear Regression for Wind Energy Forecasting

- Black circle “O”: the scatter plotting of (wind speed, wind power) pairs for the training data
- We assume linear relationship between wind power and wind speed. Then, we use linear regression to demonstrate their relationship.
- **Q:** is linear relationship a good model?

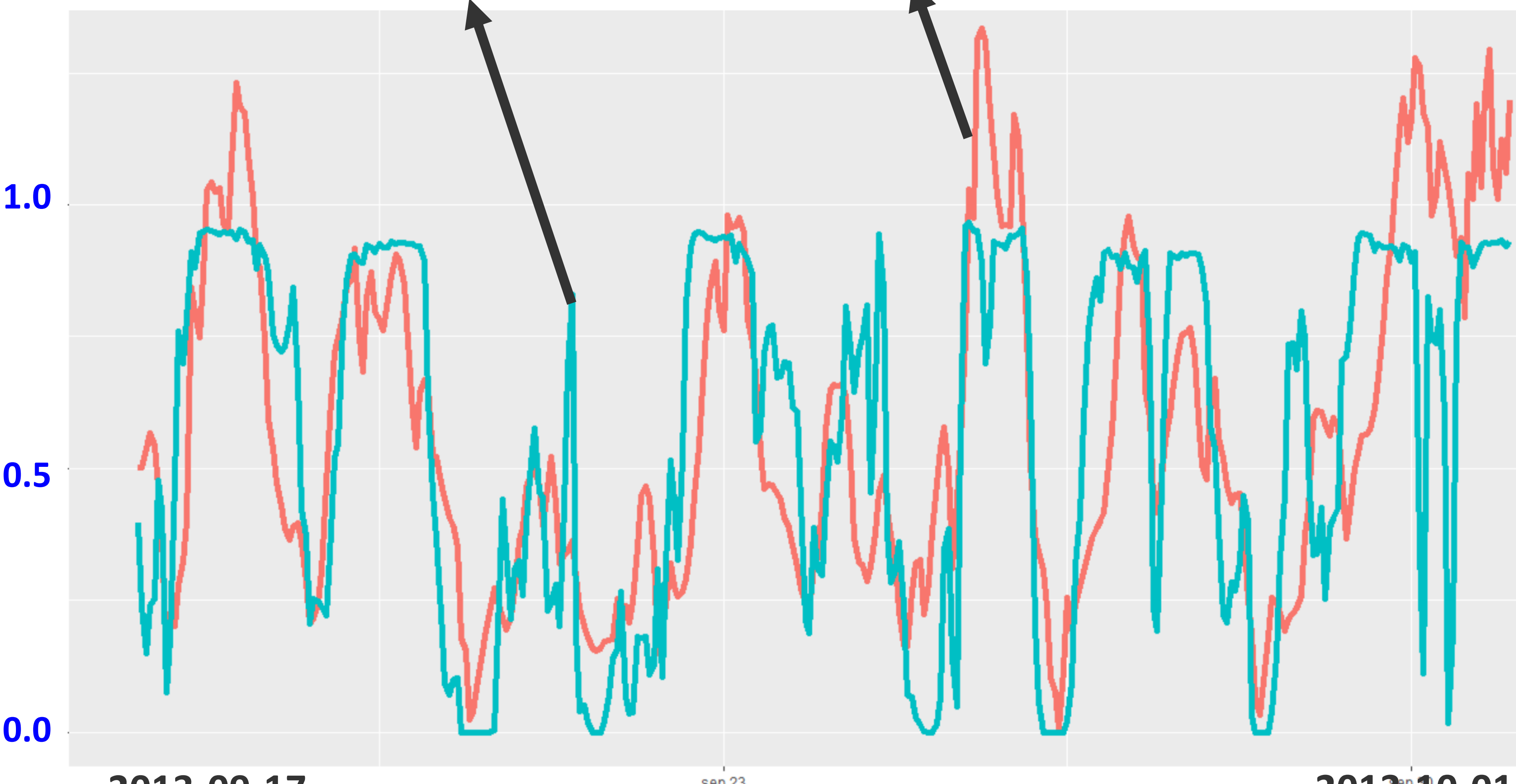
Power=0.12333 * Speed – 0.08603



True value & Predicted Wind Power for the Test Data

Predicted: Linear Regression (red line)

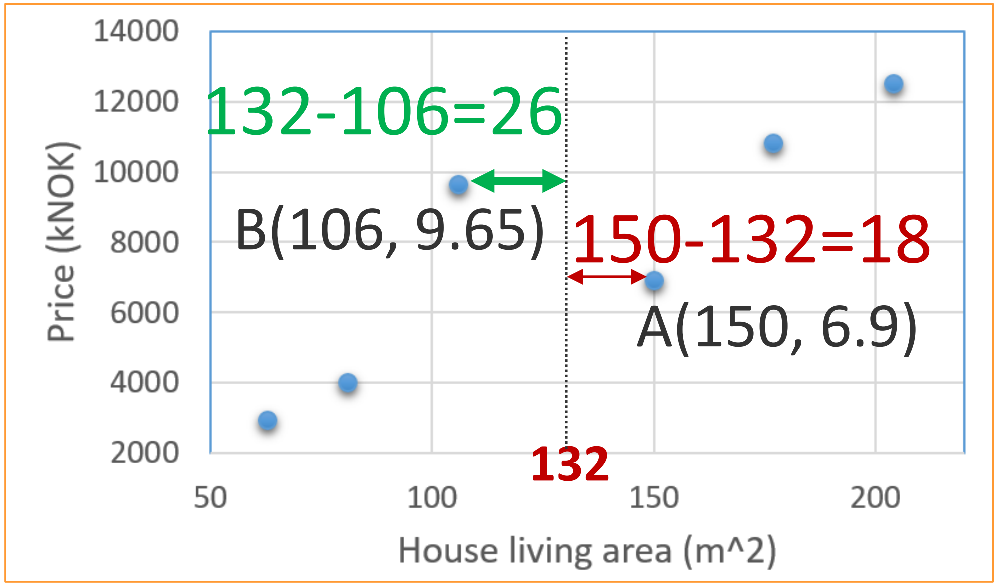
True value (blue line)



2013-09-17 01:00:00 → 1 hour scale → 2013-10-01 00:00:00

KNN AND ITS APPLICATION FOR WIND ENERGY FORECASTING

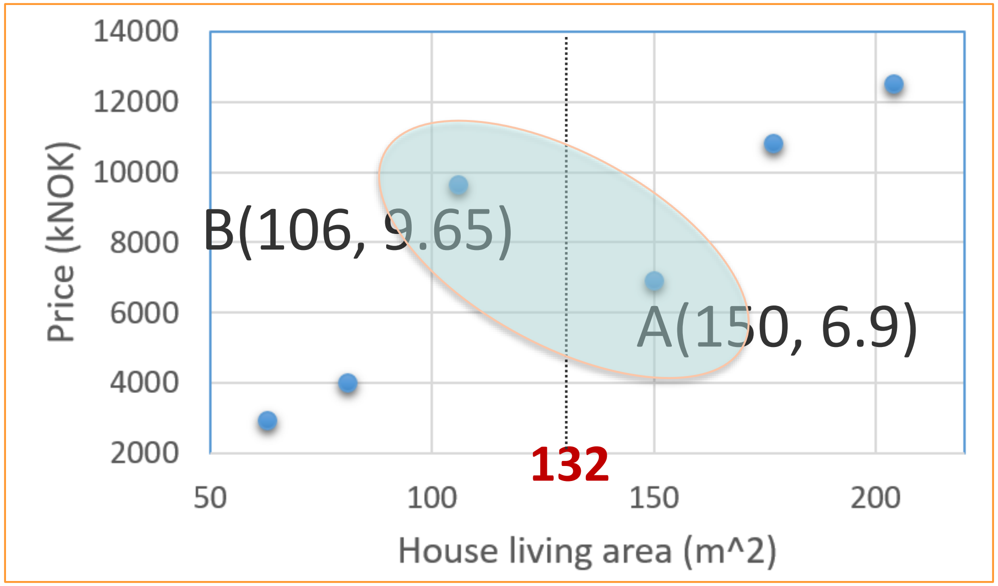
k-Nearest Neighbor (kNN) Regression. Here, ‘k’ is a variable, k=1,2,3….



- **Q: what is the price for a new house with area 132 m^2?**

- **1-NN (one nearest neighbor)**
  - **the observation A(150, 6.9) is the nearest point from the area 132 m^2**
  - **then, the predicted price has the same price as the house with area 150 m^2, which is 6.9M**

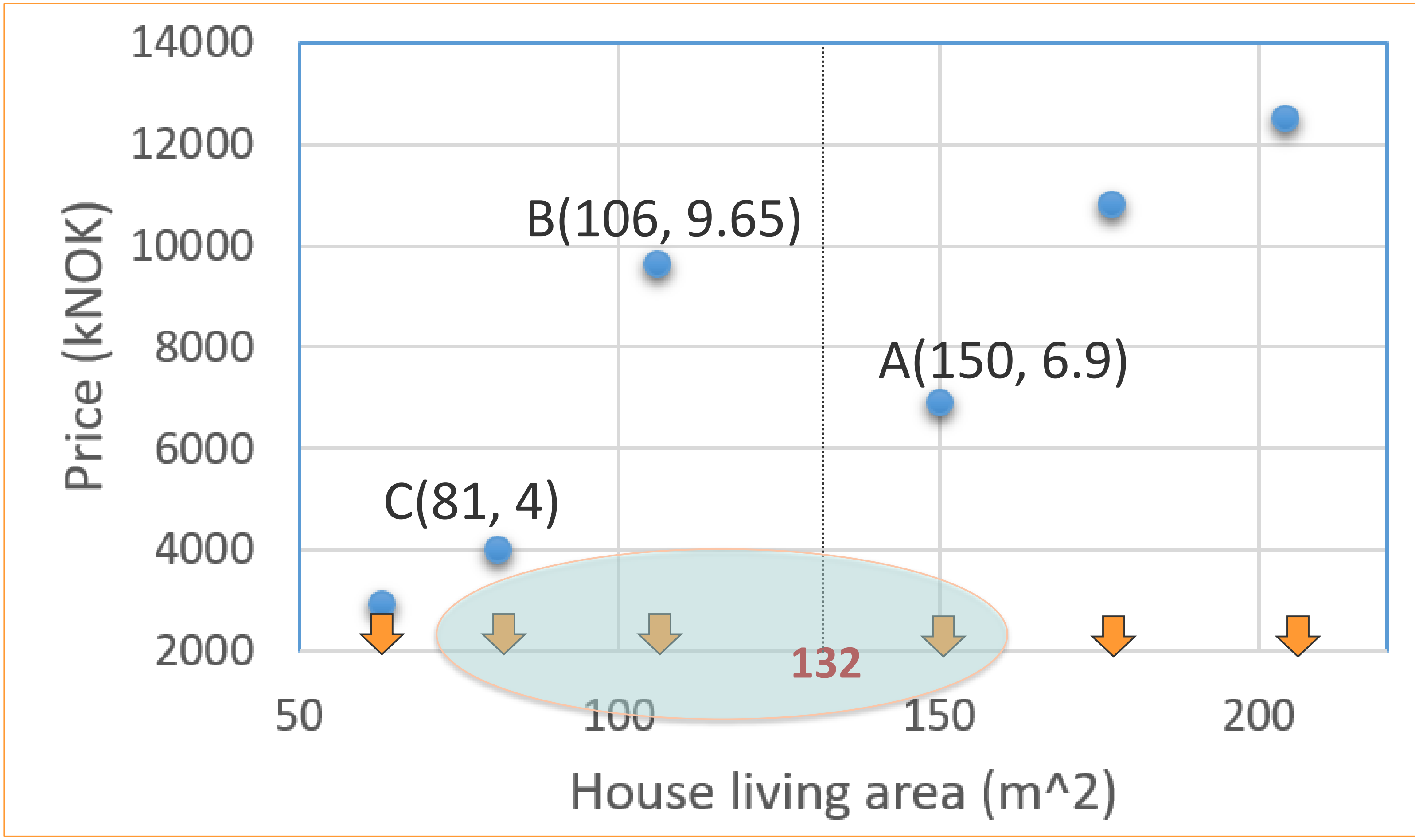
2-Nearest Neighbor (2NN)



- **Q: what is the price for a new house with area 132 m^2?**

- **2-NN (two nearest neighbors)**
  - **the observations A(150, 6.9) and B(106, 9.65) are the two nearest node from the area 132 m^2**
  - **Then, the predicted price is the average price of these two houses, i.e., (9.65+6.9)/2= 7.825M**

K-Nearest Neighbor Regression (kNN)



- **3-NN (three nearest neighbors): the observations A(150, 6.9), B(106, 9.65) and C(81, 4) are the three nearest nodes from the area 132 m^2, so, the prediction is the average of 9.65M, 6.9M and 4M, which is 6.55M**

kNN Algorithm

Given:

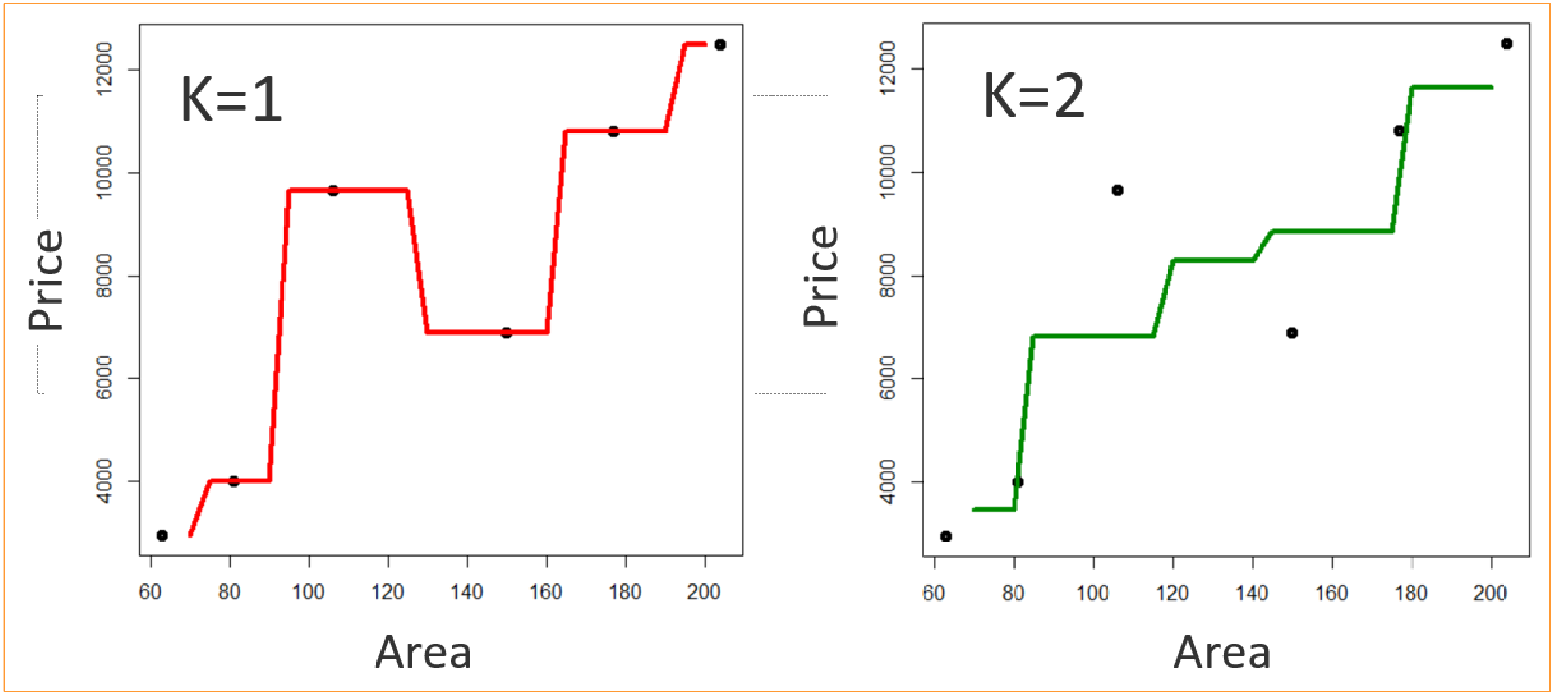
- **Training set** $(x_i, y_i), (i = 1, 2 \ldots N)$
  - x_i: **attribute-value representation of examples**
  - y_i: **real-valued target (e.g., price, rating on YouTube, profit etc)**
- **Testing point** x **that we want to predict the target**

Algorithm

- **Compute the distance** $D(x, x_i)$ **to every training example** x_i: $D(x, x_i) = |x - x_i|$
- **Select** k **closest points** $(x_{i1}, x_{i2}, \ldots, x_{ik})$ **and their values** $(y_{i1}, y_{i2}, \ldots, y_{ik})$
- **Output the average value of** $y_{i1}, y_{i2}, \ldots, y_{ik}$:

$$\overline{y} = \frac{\sum_{j=1}^{k} y_{ij}}{k}$$

kNN in R language and illustration




```
MyData <- read.csv("MyData.csv", sep=";", header=TRUE)

AreaTest = seq (70, 200, by=5); #test data set

#kNN regression on the training set

knnmodel = knn.reg(train=matrix(MyData$Area,ncol=1), test=matrix(AreaTest,ncol=1),
y=MyData$Price, k=1)

plot(MyData$Area, MyData$Price, lwd=4, xlab="Area", ylab="Price")

lines(AreaTest, knnmodel$pred,col="red",lwd=4)
```

kNN regression in R

kNN for Wind Energy Forecasting

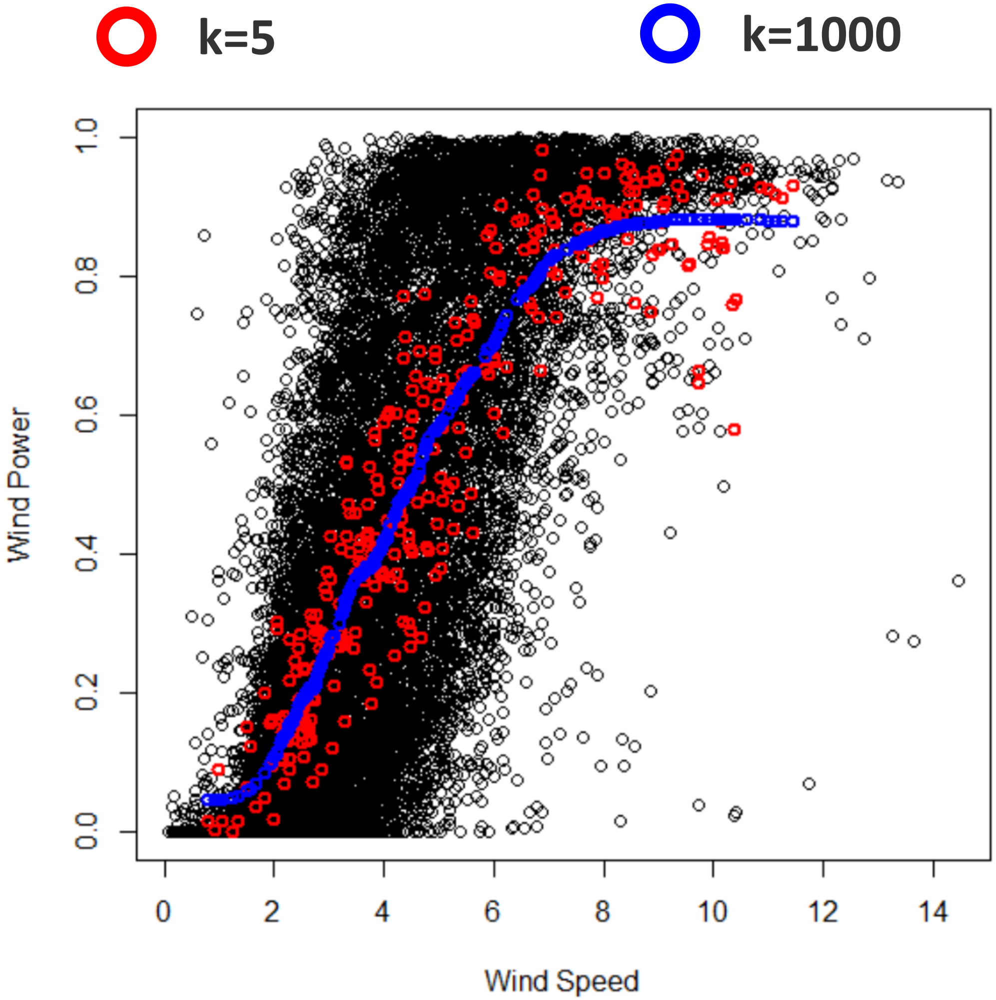
- Same real wind power data for a wind farm in Australia
- For 15336 data records, we divide them into two parts. The first 15000 data records are used as training data to build kNN model. The wind speed in the rest 336 records are used as test data to predict the wind power generation

```
powerAll = data$POWER #all wind power data set

wsAll = data$WS10 #all wind speed data set

trainLength = 15000 #number of data records to be used in training set

testLength = length(wsAll) - trainLength

wsTrain = wsAll[1:trainLength] #wind speed - training data set

powerTrain = powerAll[1:trainLength] #wind power - training data set

wsTest = wsAll[trainLength+1 : length(wsAll) ] #wind speed - test data set

wsTest = wsTest[1:testLength] #wind speed - training data set

knnmodel =
knn.reg(train=matrix(wsTrain,ncol=1),test=matrix(wsTest,ncol=1),y=powerTrain, k=10)
```

kNN regression

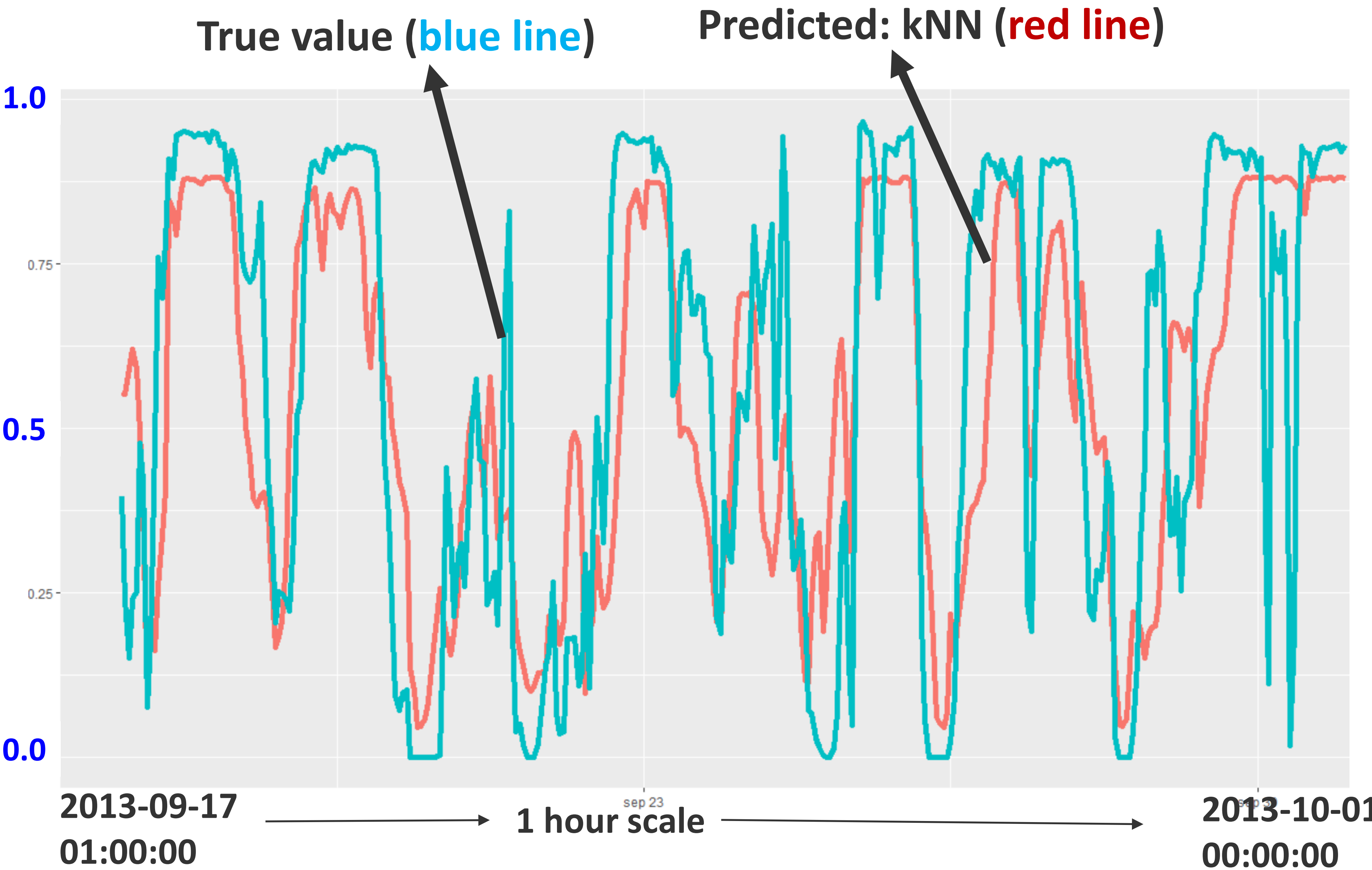
kNN for Wind Energy Forecasting

- Black circle “O”: the scatter plotting of (wind speed, wind power) pairs in the training set
- Red circle “O”: (wind speed, predicted wind power) when k=5
- Blue circle “O”: (wind speed, predicted wind power) pairs when k=1000
- In this kNN model, the curve is not linear
- Q: what is an appropriate k?



- test multiple k-values to decide an optimal value for your data
- use cross-validation to determine k. More information: http://genomicsclass.github.io/book/pages/crossvalidation.html

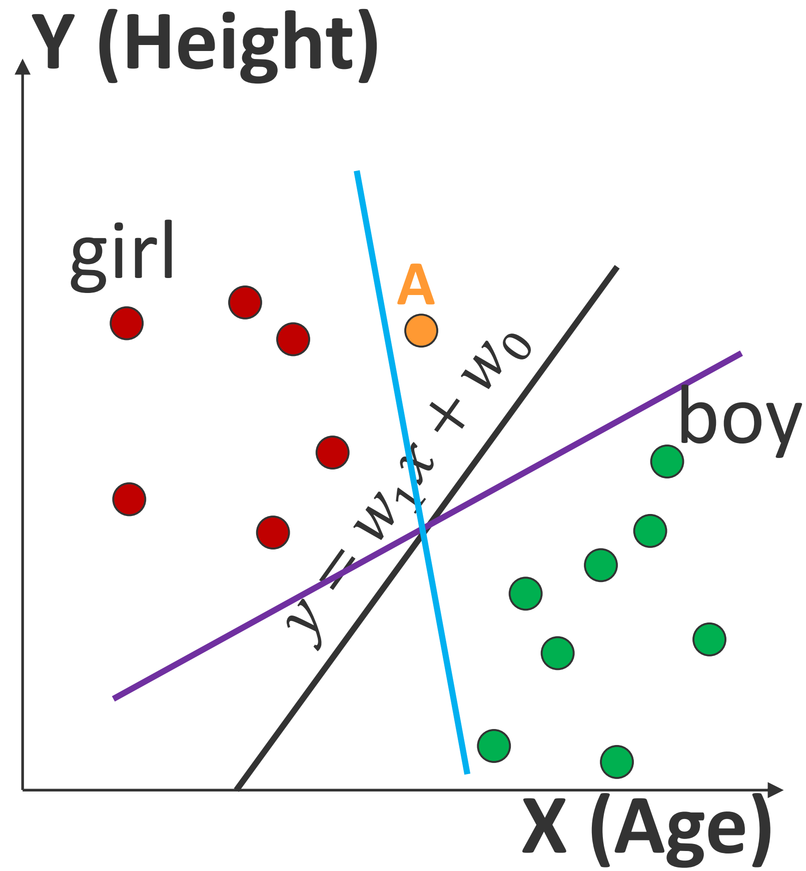
True value & Predicted Wind Power



SVR AND ITS APPLICATION FOR WIND ENERGY FORECASTING

Support Vectors Machine: an example

- **Supported Vectors Machine (SVM) can be used for classification and regression. Let's see an example of classification to understand the main idea.**
- **Goal: draw a line (called as *hyperplane*) that classifies all data in two classes.**
- **Q: Three lines in the right, which line shall we choose?**



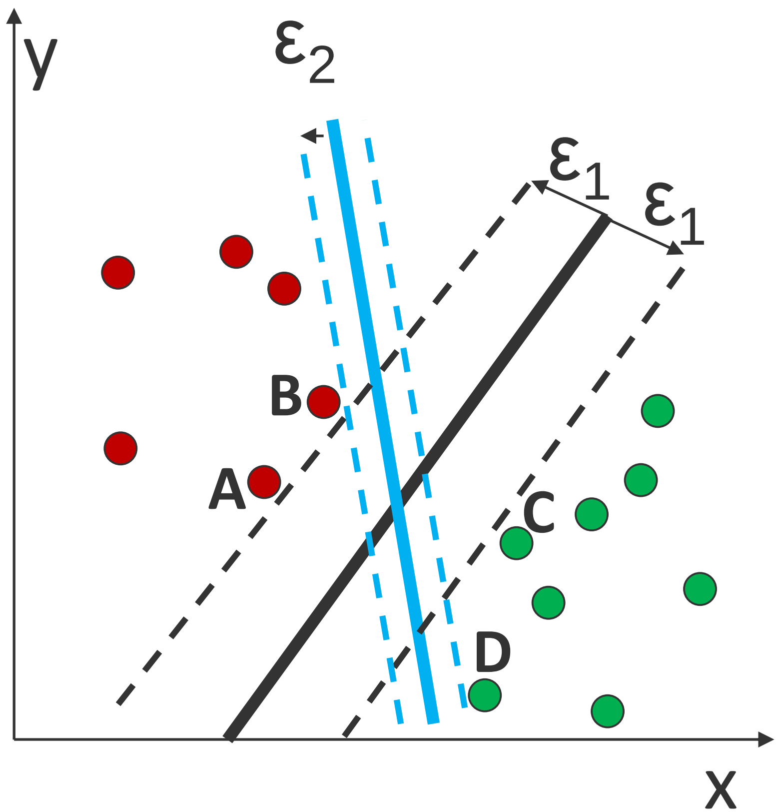
- **Observation: for new point A, it should be reasonably classified as a girl. However it will be classified incorrectly by using blue line.**
- **Intuition: find the distance between two datasets and put the line in the middle. This will reduce uncertain decisions for both classes.**

Support Vectors Machine: main idea

- **Margin**: the distance between the hyperplane and the closest nodes to the hyperplane

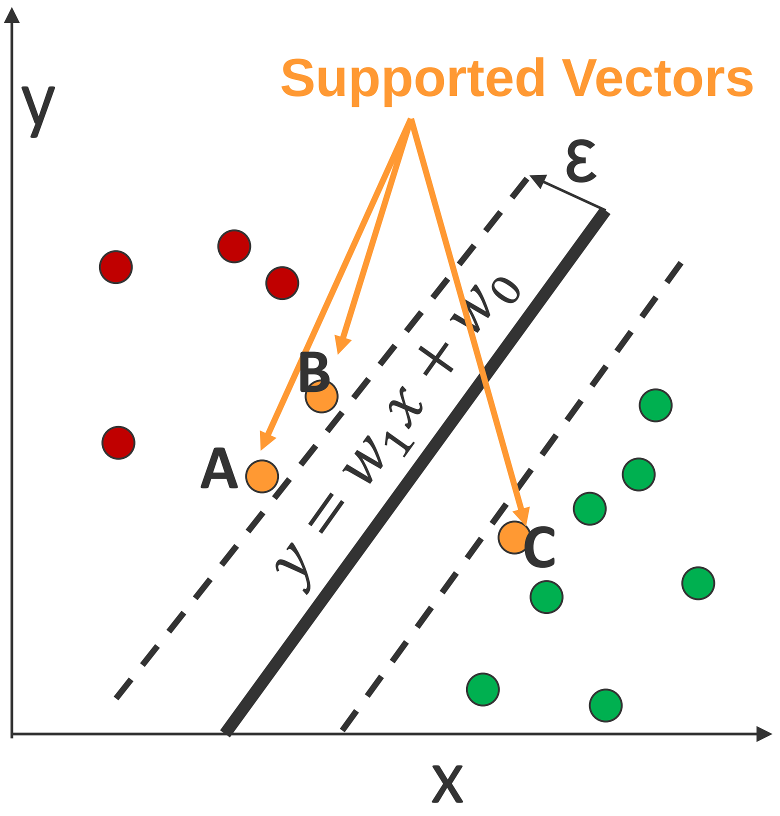
- Margin for two hyperplanes:
  - ϵ_1 for the black hyperplane and the nodes A,B and C
  - ϵ_2 for the blue hyperplane and the nodes B and D

- Clearly: $\epsilon_1 > \epsilon_2$. In this case, the choice is the black hyperplane. Large margin makes low uncertain decisions such that a slight error in measurement will not cause a misclassification.



Support Vectors Machine: main principle

- In this example, there are **three support vectors A,B,C** in orange color. These three support vectors can determine the hyperplane $y = w_1 x + w_0$.



Main principle in SVM

- The hyperplane model is determined by only a subset of the training data (i.e., the support vectors).

- The hyperplane model does not care about any training data beyond the margin ***ε***.

SVM for Regression Problem: Supported Vector Regression

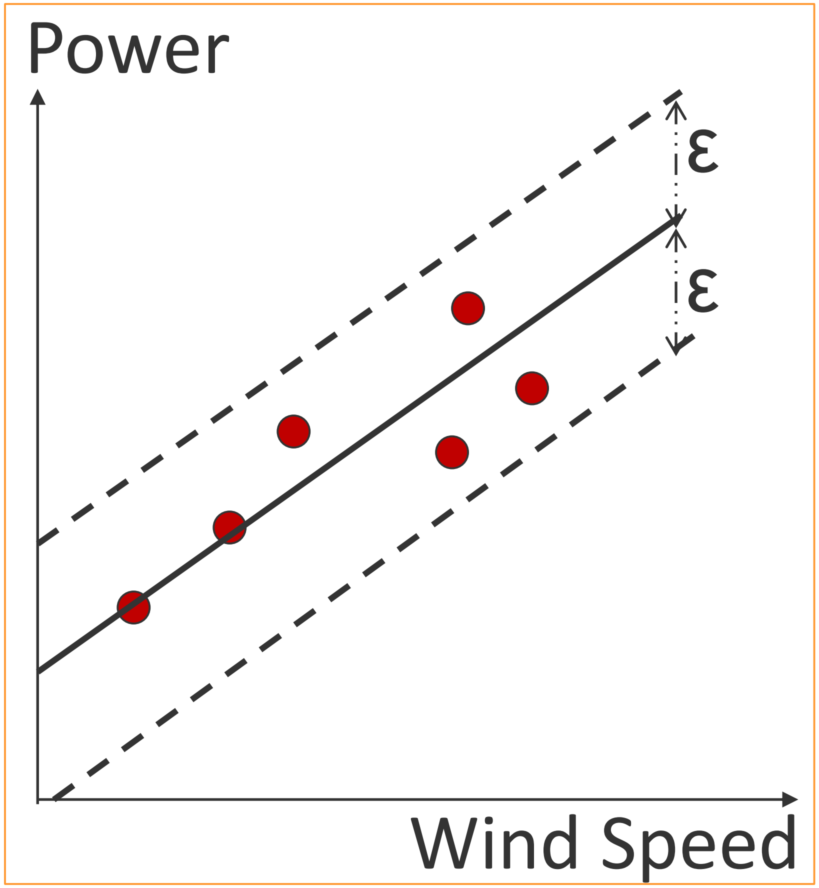
- **Supported Vector Regression (SVR): SVM is used for regression. Wind energy forecasting is a regression problem.**

- **We need to find a hyperline to show the relationship between wind power and wind speed; and then make prediction.**

Main idea in SVR:

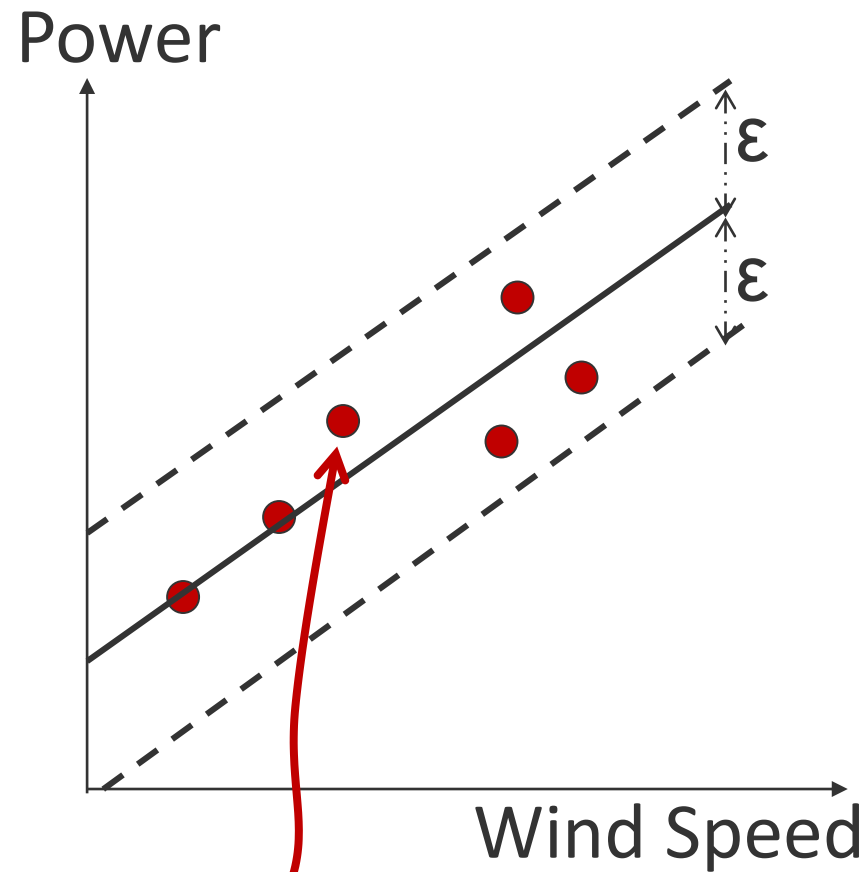
- **Margin: we set a margin *ε* which is similar as the concept in SVM.**

- **We ignore the errors as long as they are less than ε, but will not accept any deviation larger than this.**



ε-intensive error: concept

- **ε-intensive error:** **error is zero if the points are inside the band. Only the points out of the band will be considered.**
- **In R programming, SVR uses default value *ε=0.1***
- **In SVR, we draw a line such that the distance between the points and the line is not be higher than *ε*.**
- **The line is in the middle of the set of points such that the line is as close to the points as possible.**



We donot care about errors as long as they are less than ε.

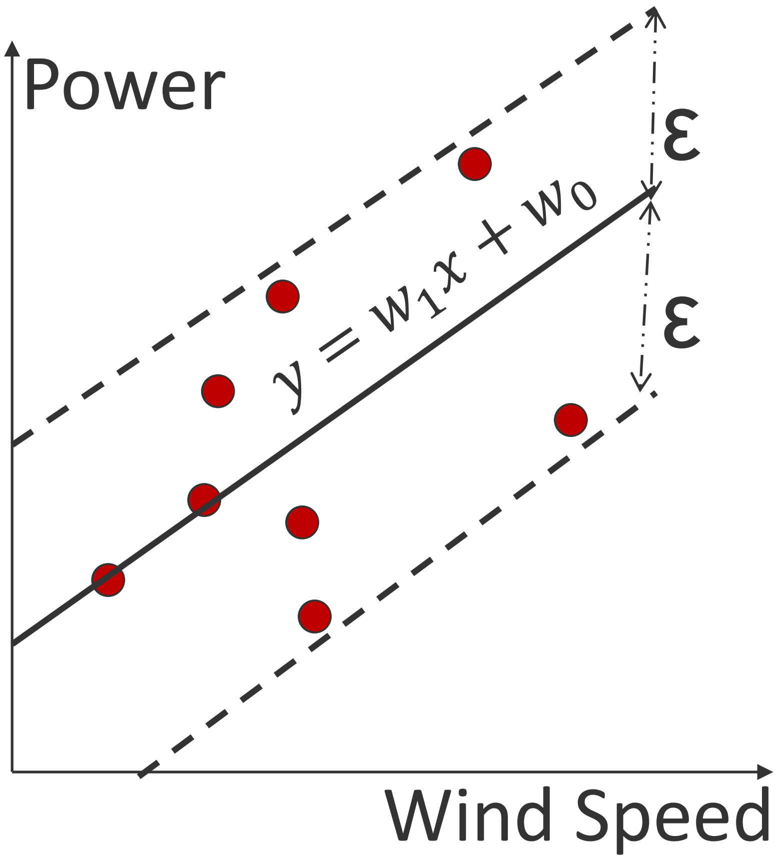
Support Vectors Regression (SVR)

- *Goal*: to find a function with at most ε-intensive error from the actual value y_i for all training data; and at the same time is as flat as possible.

- For function $y = w_1 x + w_0$, flatness means that one seeks a small w_1

$$\min \frac{1}{2} {w_1}^2$$

$$s.t.\, y_i - (w_1 x_i + w_0) \le \varepsilon$$
$$(w_1 x_i + w_0) - y_i \le \varepsilon$$



Q: why flatness?

To avoid overfitting problem, we want the function to be as flat as possible. Then, the function is less sensitive to the change of x. (Overfitting: the model does a perfect job of fitting the training data, but will do a bad job of predicting new data)

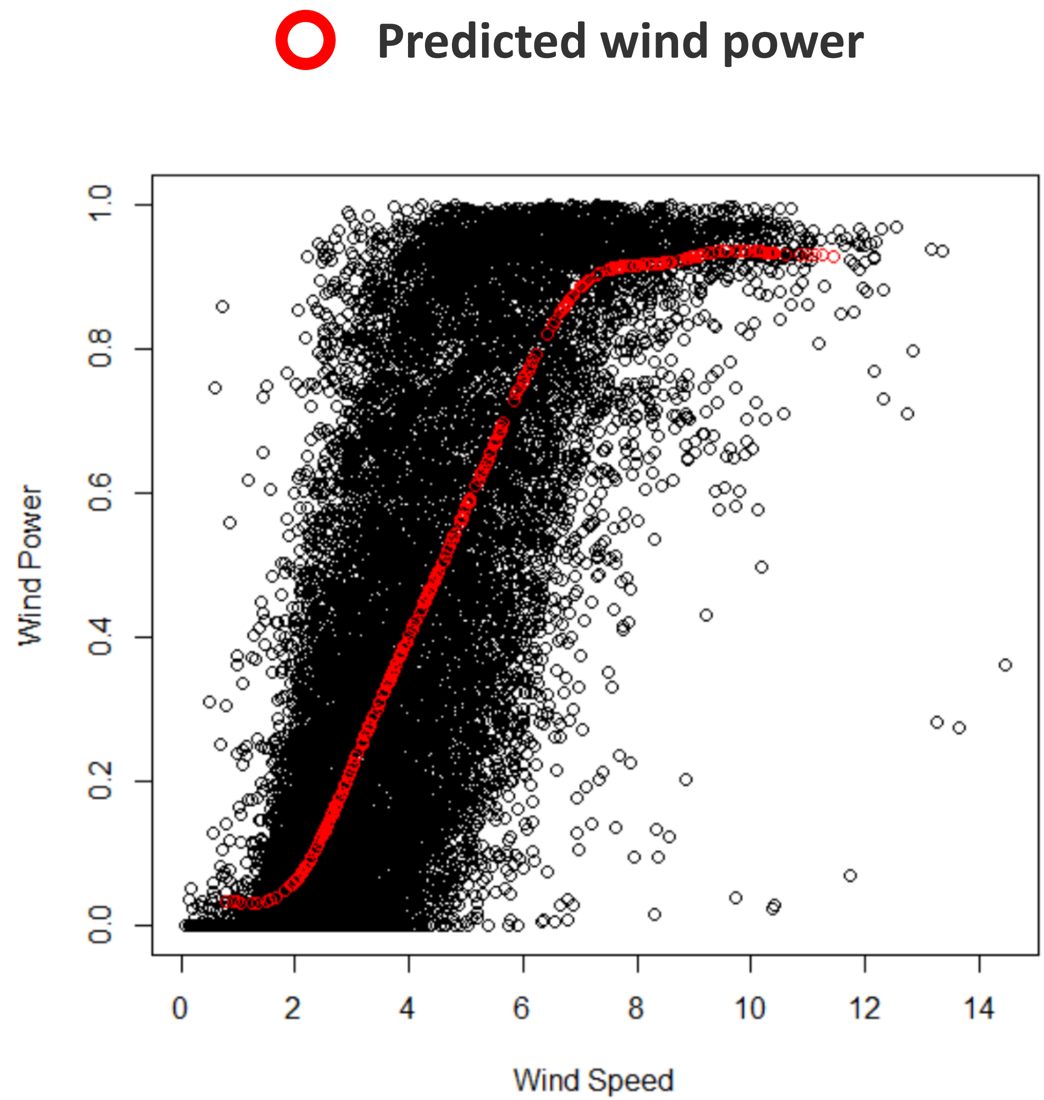
SVR for Wind Energy Forecasting

- We use the same real wind power data for a wind farm in Australia
- For all 15336 data records, we divide them into two parts. The first 15000 data records are used as training data to build SVR model. The wind speed in the rest 336 records are used as test data to find the wind power generation

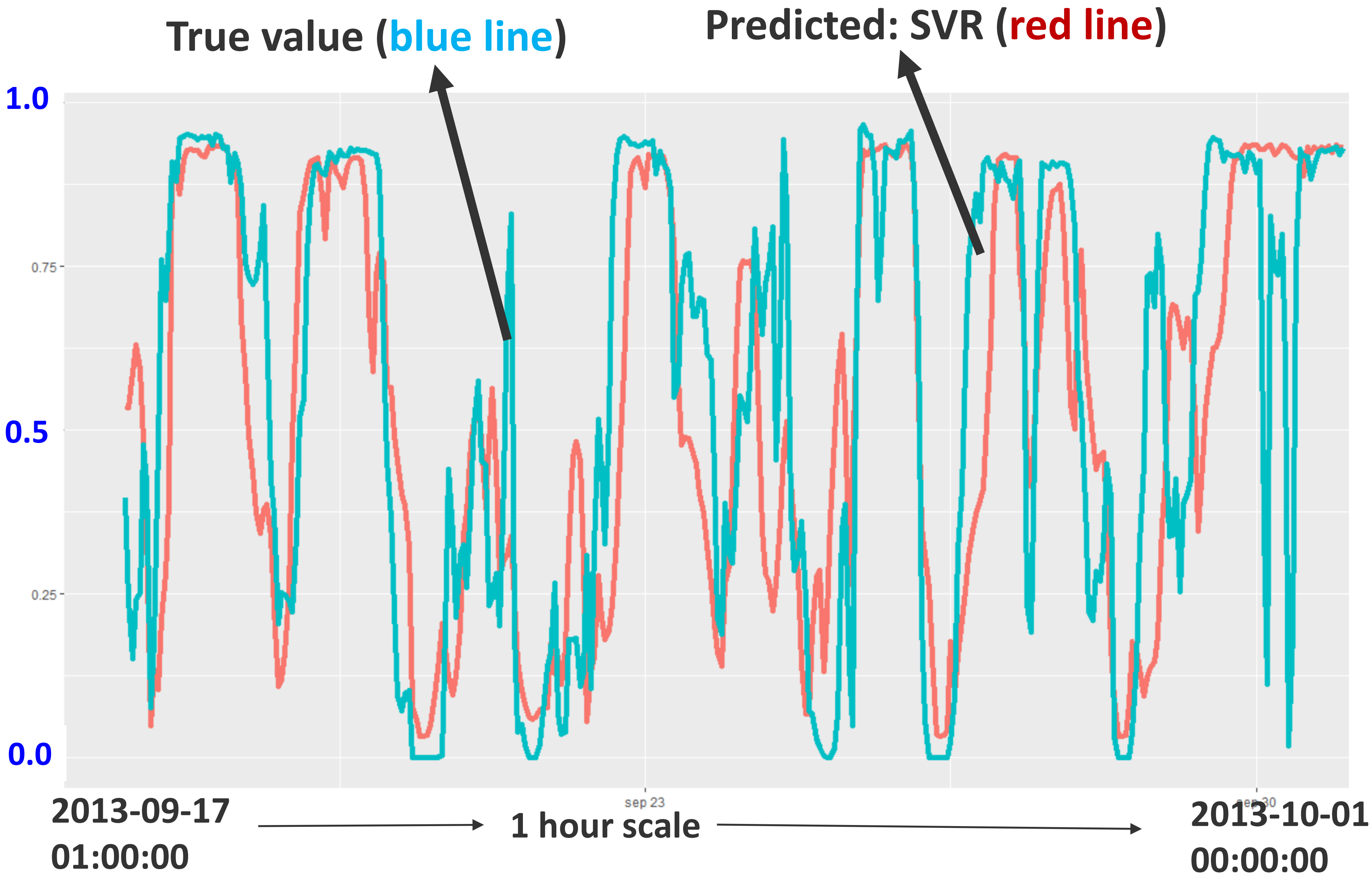
```
#use the same source code to get wsTrain, powerTrain, wsTest

plot(wsTrain, powerTrain, xlab="Wind Speed", ylab="Wind Power")

#Support vector machine/regression

df <- data.frame(x = wsTrain, y = powerTrain)

svrmodel <- svm(y ~ x, data = df)

powerPredicted = predict(svrmodel, newdata = data.frame(x = wsTest))

# Add points for fitted svrmodel

points(wsTest, powerPredicted , col = "red" )
```

SVR for Wind Energy Forecasting

- Black circle “O”: the scatter plotting of (wind speed, wind power) pairs in the training set
- Red circle “O”: (wind speed, predicted wind power) according to the wind speed in the test dataset
- Similar as kNN model, SVR-model based predicted wind power also follow a non-linear curve



True value & Predicted Wind Power for Test Data



Metrics measures the model goodness in prediction

- **Q:** we discussed three models: linear regression, kNN and SVR. Which model is the best?
- For each data point x_i, its real value is y_i. A model makes a prediction $\hat{y}_i$. In order to measure how good a model is we will compute how much error it makes. We compare each real value y_i with its predicted value $\hat{y}_i$ and calculate the difference.
- The expression $\hat{y}_i - y_i$ is the error. For each data point, we have the sum of the squared errors, and then take the mean. We have the Mean Squared Error (MSE)

$$MSE = \frac{1}{N}\sum_{i=1}^{N}(\hat{y}_i - y_i)^2$$

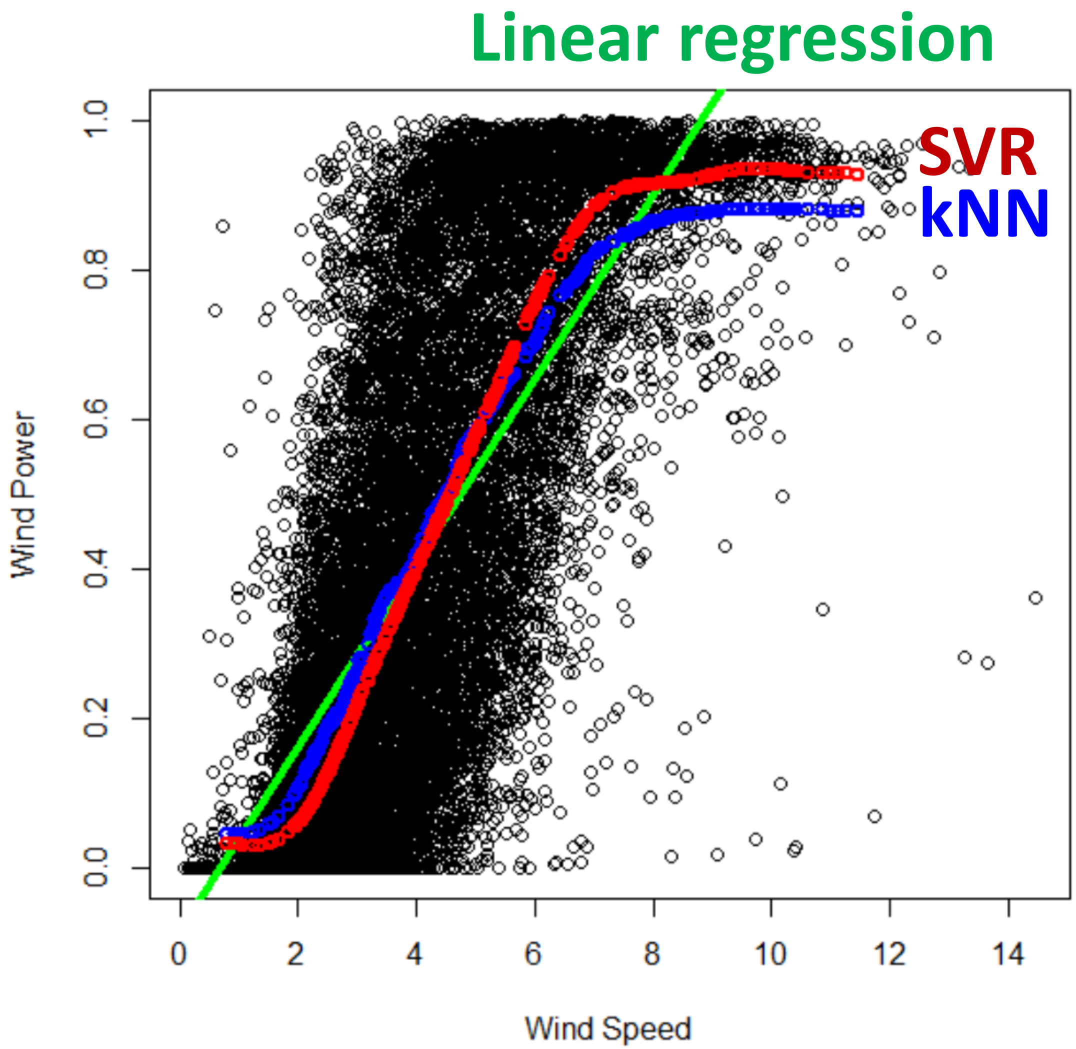
- A common way to measure error uses the Root Mean Squared Error (RMSE), we take the square root of MSE

$$RMSE = \sqrt{MSE}$$

An illustration to show the RMSE between these models

```
rmse <- function(error)

{     sqrt(mean(error^2))

}

# Calculate error

errorSVR <- powerTest -
powerPredicted

rmse(errorSVR)
```

Model	RMSE
Linear regression	0.2545
kNN (k=1000)	0.2276
SVR	0.2343



MORE CONSIDERATIONS…

Different time horizon prediction

- **Normally we need short-term forecasting for solar or wind energy power generation**
- **Q: in what cases, we need very short-term forecasting, e.g., 5mins?**

Solar power plant in desert

More parameters

- House price is not only dependent on the area. It is also related to location, public transport, crime statistics, school rankings, etc. How to use machine learning to model and solve this problem?

- For example, if we consider more factors affecting house price, we have **Multiple Linear Regression** model

$$\text{Price} = w_0 + w_1 * \text{Area} + w_2 * \text{Location} + w_3 * \text{SchoolQuality}$$

- Similarly, wind power generation is not only dependent on wind speed. It is also related to wind direction, temperature, and pressure.

$$\text{Wind Power} = w_0 + w_1 * \text{WindSpeed} + w_2 * \text{WindDirection} + w_3 * \text{Temperature} + w_4 * \text{Pressure}$$

Code:
```
lm(powerTrain ~ wsTrain + winddirection + temperature + pressure)
```