Lecture Objectives

- Understanding the outputs of a Linear Model
- Limitations of Linear Models and their extensions
- How to reduce over-fitting/variance via regularization
- Support Vector Machines (SVM)

Brief review of the linear model

$$Y = f(X) + \varepsilon$$

response — predictor — irreducible error

If we assume that y is a linear combination of the predictor variables,

$$\hat{y} \cong \hat{\beta}_0 + \hat{\beta}_1 x_1 + \hat{\beta}_2 x_2 + \cdots + \hat{\beta}_p x_p$$

We have n observations (training data);

$$y_1 \cong \hat{\beta}_0 + \hat{\beta}_1 x_{11} + \hat{\beta}_2 x_{12} + \hat{\beta}_3 x_{13} + \cdots + \hat{\beta}_p x_{1p}$$

$$\begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{pmatrix} \cong \begin{pmatrix} 1 & x_{11} & x_{12} & \cdots & x_{1p} \\ 1 & x_{21} & x_{22} & \cdots & x_{2p} \\ \vdots & \vdots & \vdots & \cdots & \vdots \\ 1 & x_{n1} & x_{n2} & \cdots & x_{np} \end{pmatrix} \begin{pmatrix} \hat{\beta}_0 \\ \hat{\beta}_1 \\ \hat{\beta}_2 \\ \vdots \\ \hat{\beta}_p \end{pmatrix}$$

$$Y \cong X \qquad \beta$$

Our goal is to minimize $\|Y - X\beta\|_2$

Possible Questions

- How accurately do we know our model parameters?
- Is at least one predictor variable useful in the prediction?
  - We have to examine the p-values
- Which subset of the predictor variables are important?
  - There are several techniques of predictor variable/feature selection
- What would be the accuracy of predictions on unseen data?
  - We can generate confidence intervals on our estimates
  - Cross-validation gives us an estimate.
- Do I need more predictor variables/features?
  - Look at patterns in the residual errors

Confidence intervals for predictor estimators

- What causes errors in estimation of $\boldsymbol{\beta_i}$?
  - $\boldsymbol{\varepsilon}$ (noise in the model)
  - we do not know the exact form of $f(x)$
  - limited sample size
- Variance of $\widehat{\beta}_i$ is called as ***standard error***, $SE(\hat{\beta}_i)$
- To estimate SE, we use **Bootstrapping**
  - sampling from the training data *(X,Y) to* estimate its statistical properties.
- In our case, we can sample with replacement
  - Compute $\hat{\beta}_i$multiple times by random sampling
  - Variance of multiple estimates approximates the true variance

Standard Errors Intuition from Formulae

- **Better model:** $(\hat{f} - y_i) \downarrow \Longrightarrow \sigma \downarrow \Longrightarrow SE \downarrow$

$$\sigma \approx \sqrt{\sum \frac{(\hat{f}(x) - y_i)^2}{n-2}}$$

- **More data:** $n \uparrow$ and $\sum_i (x_i - \bar{x})^2 \uparrow \Longrightarrow SE \downarrow$
- **Larger coverage**: $var(x)$ or $\sum_i (x_i - \bar{x})^2 \uparrow \Longrightarrow SE \downarrow$
- **Better data:** $\sigma^2 \downarrow \Rightarrow SE \downarrow$

$$SE\left(\widehat{\beta}_0\right) = \sigma\sqrt{\frac{1}{n} + \frac{\overline{x}^2}{\sum_i (x_i - \overline{x})^2}}$$

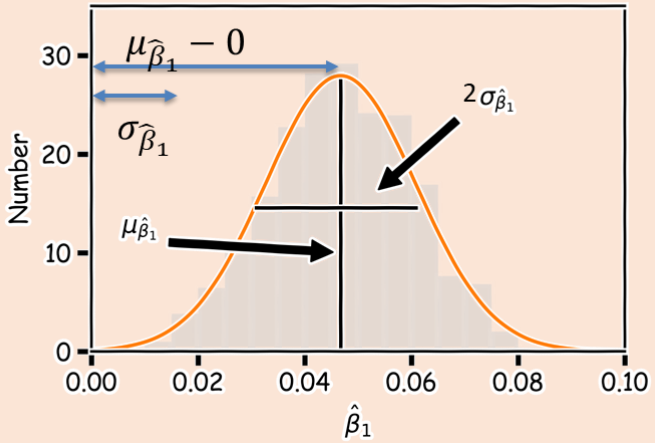
$$SE\left(\widehat{\beta}_1\right) = \frac{\sigma}{\sqrt{\sum_i (x_i - \overline{x})^2}}$$

$$General\ Formula\colon\ SE(\beta)^2 = \sigma^2 (X^T X)^{-1}$$

Significance of predictor variables

- As we saw, there are inherent uncertainties in estimation of **β**
- We evaluate the importance of predictors using hypothesis testing, using the t-statistics and p-values (Small p-value(<0.05) => significant)
- Null hypothesis is that $\beta_i=0$

Test statistic here would be $t = \frac{\mu_{\hat{\beta}_1}}{\sigma_{\hat{\beta}_1}}$
Which measures the distance of the mean from zero in units of standard deviation.



Subset Selection Techniques

- Total number of subsets of a set of size J = ?
- **Goal:** All the variables in the model should have sufficiently low p-values, and all the variables outside the model should have a large p-value if added to the model.
- Three possible approaches
  - Forward selection
  - Backward selection
  - Mixed selection

Subset Selection Techniques

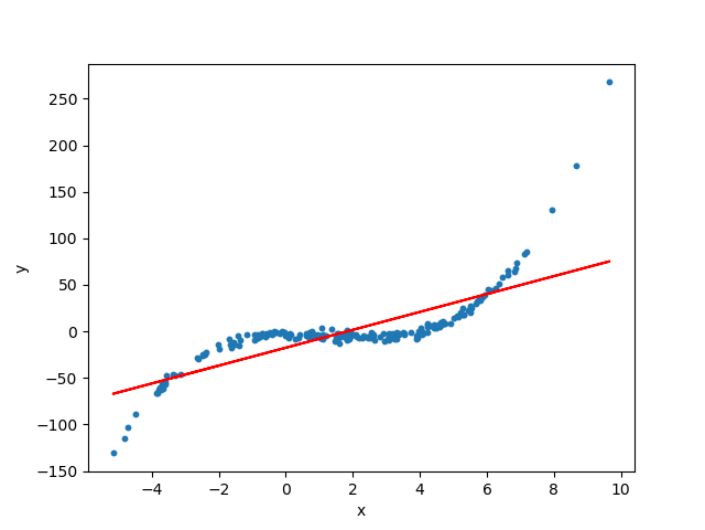
- **Forward selection:**
  - Begin will a null set, S
  - Perform J linear regressions, each with exactly one variable
  - Add the variable that results in lowest Cross-validation error to the set, S
  - Again, perform J-1 linear regressions with 2 variables
  - Add the variable that results in lowest Cross-validation error to the set, S
  - Continue until some stopping criteria is reached… eg. CV error is not decreasing
- **Backward selection** begins with all the variables and removes the variable with highest p-value at successive steps
- **Mixed selection** is similar to Forward Selection, but it may also remove a variable if it doesn't yield any improvement to the model

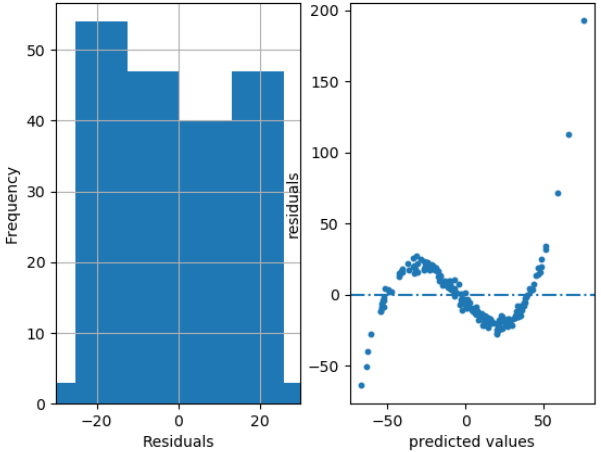
Do I need more predictors/change of model?

- When we estimated the variance of ϵ, we assumed that the residuals $r_i = y_i - \hat{y}_i$ were uncorrelated and normally distributed with mean 0 and fixed variance.
- These assumptions need to be verified using the data. In residual analysis, we typically create two types of plots:

1. a plot of r_i with respect to x_i or $\hat{y}_i$. This allows us to compare the distribution of the noise at different values of x_i.
2. a histogram of r_i. This allows us to explore the distribution of the noise independent of x_i or $\hat{y}_i$.

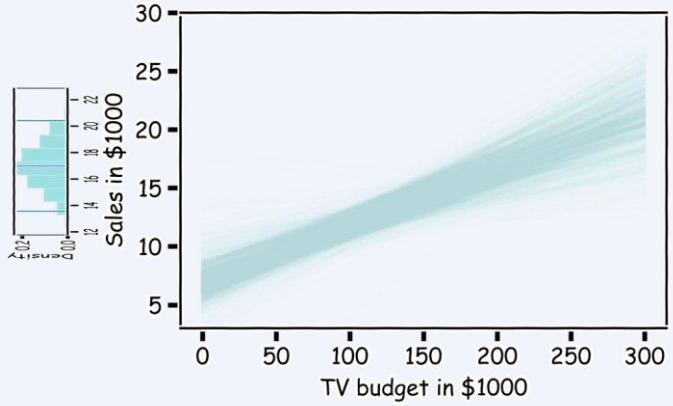
Patterns in Residuals

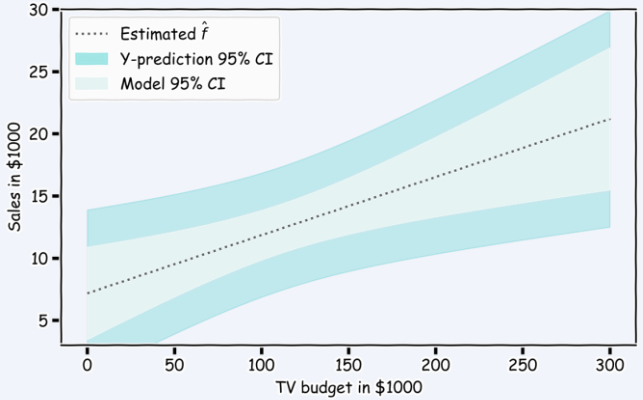




- Residuals are easier to interpret than the model
- We plot $(y_i - \hat{y}_i)$ with y_i, so the graph is always 2-D

Confidence intervals on predictions of y





- Depends on confidence on **β**
- Different **β** => different values of y
- Given x, examine distribution of $\hat{f}$, determine the mean and standard deviation.
- For each of these $f(x)$, the prediction for $y \sim N(f, \sigma_\epsilon)$

Potential problems of Linear Models

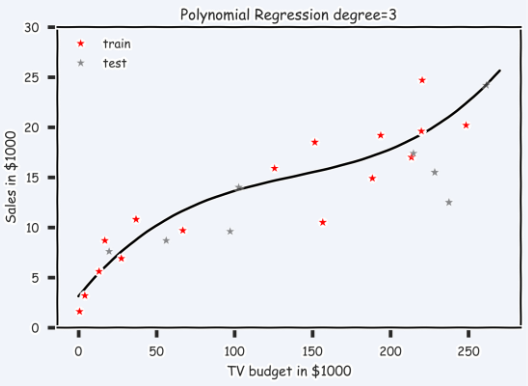
- Non-linearity
  - Can use polynomial linear regression or design better features
- Outliers
  - Disturbs the models because of quadratic penalty, Discard outliers carefully
- High-leverage points
  - Outliers in the predictor variables
- Collinearity (2 or more predictor variables have high correlation)
  - Keep one of them or design a good combined feature
- Correlation of error terms, Non-constant variance of error terms
  - Gives higher confidence in the model, can't trust the CI on model parameter

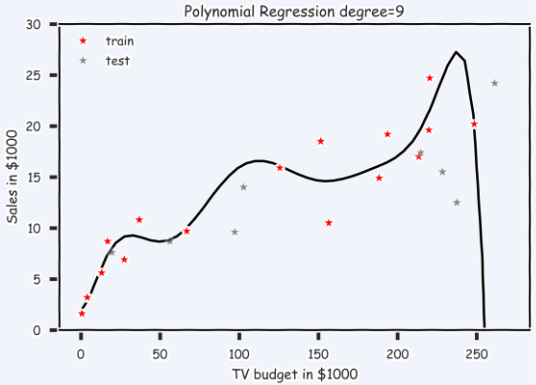
Polynomial Regression

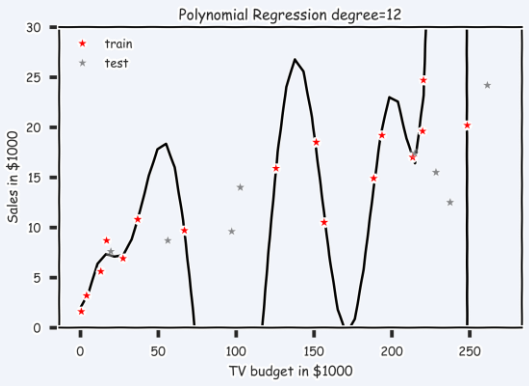
- The simplest non-linear model we can consider, for a response *Y* and a predictor *X*, is a polynomial model of degree *M*, $y = \beta_0 + \beta_1 x + \beta_2 x^2 + \ldots + \beta_M x^M + \epsilon.$
- Just as in the case of linear regression with cross terms, polynomial regression is a special case of linear regression - we treat each x^m as a separate predictor. Thus, we can write

$$\mathbf{Y} = \begin{pmatrix} y_1 \\ \vdots \\ y_n \end{pmatrix}, \qquad \mathbf{X} = \begin{pmatrix} 1 & x_1^1 & \ldots & x_1^M \\ 1 & x_2^1 & \ldots & x_2^M \\ \vdots & \vdots & \ddots & \vdots \\ 1 & x_n & \ldots & x_n^M \end{pmatrix}, \qquad \boldsymbol{\beta} = \begin{pmatrix} \beta_0 \\ \beta_1 \\ \vdots \\ \beta_M \end{pmatrix}.$$

Polynomial Regression

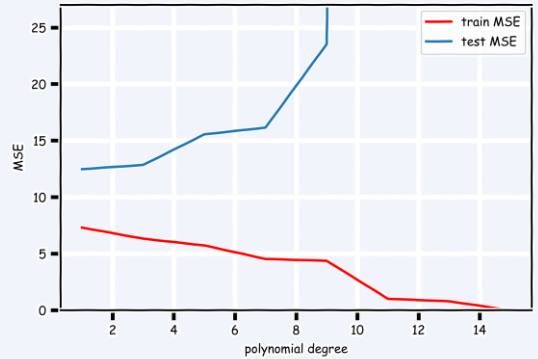






- Which of the above three is the best model?
  - Check RMSE
  - Check R^2
  - Remember bias and variance??

Benefit of Cross-Validation



$$CV(\text{Model}) = \frac{1}{K}\sum_{i=1}^{K} L(\hat{f}_{C_{-i}}(C_i))$$

- Using cross-validation, we generate validate the models on a portion of training data which our learning algorithm has never seen.
- Leave-one out method is used when the number of sample points is very small.

Regularization of Linear Models

- Goal: Reduce over-fitting of the data by reducing degrees of freedom
- For a linear model, regularization is typically achieved by constraining the weights of the model

$$L_{reg}(\beta) = L(\beta) + \lambda R(\beta),$$

where λ is a scalar that gives the weight (or importance) of the regularization term.

- Fitting the model using the modified loss function L_{reg} would result in model parameters with desirable properties (specified by R).

Ridge Regression

- Alternatively, we can choose a regularization term that penalizes the squares of the parameter magnitudes. Then, our regularized loss function is:

$$L_{Ridge}(\beta) = \frac{1}{n}\sum_{i=1}^{n} |y_i - \boldsymbol{\beta}^\top \boldsymbol{x}_i|^2 + \lambda \sum_{j=1}^{J} \beta_j^2.$$

- Works best when least-square estimates have high variance
- As λ increases, flexibility decreases, variance decreases, bias increases slightly

- Note that $\sum_{j=1}^{J} |\beta_j|^2$ is the $\boldsymbol{l_2}$ norm of the vector $\boldsymbol{\beta}$ $\sum_{j=1}^{J} \beta_j^2 = \|\boldsymbol{\beta}\|_2^2$

Ridge Regression

- We often say that L_{ridge} is the loss function for l_2 regularization.
- Finding the model parameters β_{ridge} that minimize the l_2 regularized loss function is called ***ridge regression.***

In [ ]:
```
from sklearn.linear_model import Ridge
```

In [20]:
```
X_train = train[all_predictors].values
X_val = validation[all_predictors].values
X_test = test[all_predictors].values

ridge_regression = Ridge(alpha=1.0, fit_intercept=True)
ridge_regression.fit(np.vstack((X_train, X_val)), np.hstack((y_train, y_val)))

print('Ridge regression model:\n {} + {}^T . x'.format(ridge_regression.intercept_, ridge_regression.coe
```

```
Ridge regression model:
 -525.7662550875951 + [ 0.24007312  8.42566029  2.04098593 -0.04449172 -0.01227935  0.41902475
 -0.50397312 -4.47065168  4.99834262  0.          0.          0.29892679]^T . x
```

In [21]:
```
print('Train R^2: {}, test R^2: {}'.format(ridge_regression.score(np.vstack((X_train, X_val)),
                                                                  np.hstack((y_train, y_val))),
                                           ridge_regression.score(X_test, y_test)))
```

```
Train R^2: 0.5319764744847737, test R^2: 0.7881798111697319
```

LASSO (least absolute shrinkage and selection operator) Regression

- Ridge regression reduces the parameter values but doesn't force them to go to zero. LASSO is very effective in doing that.
- It uses the following regularized loss function is:

$$L_{LASSO}(\beta) = \frac{1}{n}\sum_{i=1}^{n}|y_i - \boldsymbol{\beta}^\top \boldsymbol{x}_i|^2 + \lambda \sum_{j=1}^{J}|\beta_j|.$$

- Note that $\sum_{j=1}^{J}|\beta_j|$ is the $\boldsymbol{l_1}$ norm of the vector $\boldsymbol{\beta}$

$$\sum_{j=1}^{J}|\beta_j| = \|\boldsymbol{\beta}\|_1$$

LASSO Regression

- Hence, we often say that L_{LASSO} is the loss function for l_1 regularization.
- Finding the model parameters β_{LASSO} that minimize the l_1 regularized loss

In [ ]:
```
from sklearn.linear_model import Lasso
```

In [22]:
```
lasso_regression = Lasso(alpha=1.0, fit_intercept=True)
lasso_regression.fit(np.vstack((X_train, X_val)), np.hstack((y_train, y_val)))

print('Lasso regression model:\n {} + {}^T . x'.format(lasso_regression.intercept_, lasso_regression.coe
```

```
Lasso regression model:
 10.424895873901445 + [ 0.24482603  3.48164594  1.84836859 -0.06864603 -0.         -0.
 -0.02249766 -0.          0.          0.          0.          0.        ]^T . x
```

In [23]:
```
print('Train R^2: {}, test R^2: {}'.format(lasso_regression.score(np.vstack((X_train, X_val)),
                                                                  np.hstack((y_train, y_val))),
                                           lasso_regression.score(X_test, y_test)))
```

```
Train R^2: 0.48154992527975765, test R^2: 0.6846451270316087
```

Choosing λ

- In both ridge and LASSO regression, we see that the larger our choice of the **regularization parameter** λ, the more heavily we penalize large values in **β,**
- If λ is close to zero, we recover the MSE, i.e. ridge and LASSO regression is just ordinary regression.
- If λ is sufficiently large, the MSE term in the regularized loss function will be insignificant and the regularization term will force $\boldsymbol{\beta}_{\text{ridge}}$ and $\boldsymbol{\beta}_{\text{LASSO}}$ to be close to zero.
- To avoid ad-hoc choices, we should select λ using cross-validation.
- Once the model is trained, we use the unregularized performance measure to evaluate the model's performance.

Elastic Net

- Middle ground between Ridge and Lasso regression
- Regularization term is a simple mix with parameter ‘r’

$$L_{EN}(\beta) = \frac{1}{n}\sum_{i=1}^{n}|y_i - \beta^T x_i|^2 + (1-r)\lambda\sum_{j=1}^{J}\beta_j^2 + r\lambda\sum_{j=1}^{J}|\beta_j|$$

- Elastic Net has better convergence features over Lasso.

```
Sklearn.linear_model import ElasticNet
Elastic_net = ElasticNet(alpha=0.1, l1_ratio=0.5)
```

SVM (Support Vector Machines)

- Uses a different approach to regression
- Instead of thinking of the fit as a line, let us think of it as a channel
- Fit as many instances as possible on the channel while limiting the margin violations, (i.e. instances off the channel)
- Width of the channel is the hyper-parameter ‘ε’
- Adding more training instances within the channel doesn’t change the model parameters
- Hence these models are more robust against over-fitting

SVM Regression

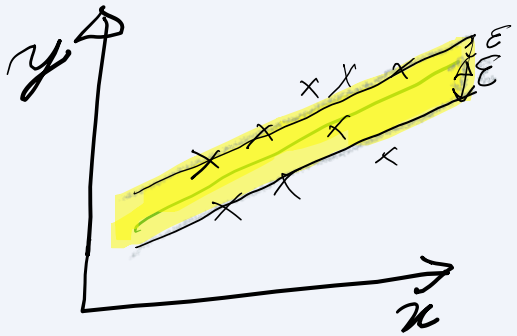
Assume $f(x) = x^T\beta + \beta_0$

Let's minimize $H(\beta, \beta_0) = \sum_{i=1}^{N} V(y_i - f(x_i)) + \frac{\lambda}{2}\|\beta\|^2$

If we consider the Optimization Problem

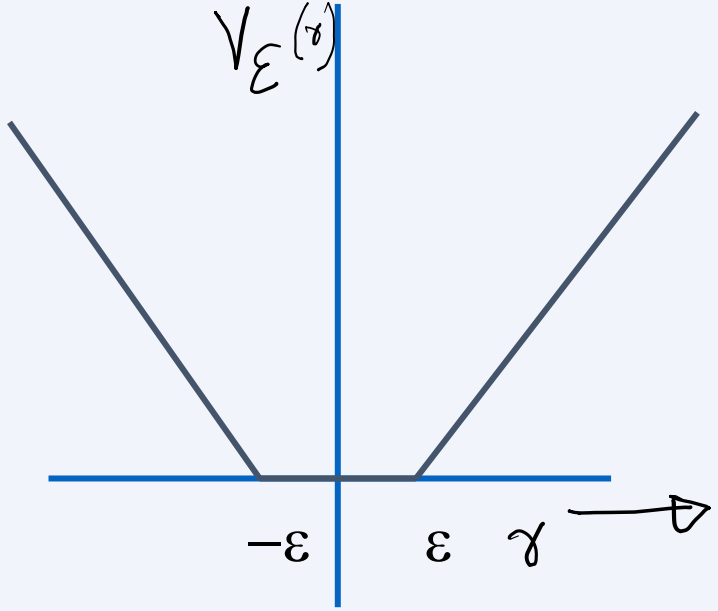
$$\text{minimize} \quad \frac{1}{2}\|\beta\|^2$$

$$\text{st} \begin{cases} y_i - f(x_i) \le \varepsilon \\ f(x_i) - y_i \le \varepsilon \end{cases}$$



For a fixed ε, we may not have a feasible solution

ε insensitive Loss function



$$V_\varepsilon(\gamma) = \begin{cases} 0 & \text{if } |\gamma| < \varepsilon \\ |\gamma| - \varepsilon, & \text{otherwise} \end{cases}$$

SVM Regression

So, let us allow some slack variables ξ_i & ξ_i^* (support vectors)

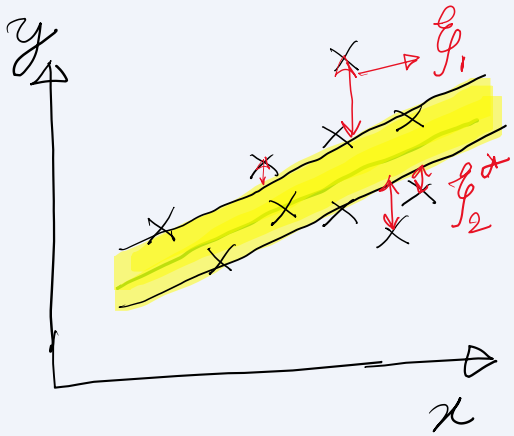
Consider the optimization problem:-

Minimize $\frac{1}{2}\|\beta\|^2 + C\sum_{i=1}^{n}(\xi_i + \xi_i^*)$

$$y_i - f(x_i) \leq \varepsilon + \xi_i$$

$$f(x_i) - y_i \leq \varepsilon + \xi_i^*$$

$$\xi_i \geq 0 \quad \xi_i^* \geq 0$$




```
Sklearn.svm import ElasticNet
Svm_reg = LinearSVR(epsilon=1, C=2)
```

Parameters in SVM regression

- Parameter ε controls the width of the channel and can affect the number of support vectors used to construct the regression function.
- Adding more training vectors
- Bigger ε => fewer support vectors
- Parameter C determines the trade-off between the model complexity and the degree to which the deviations larger than ε can be tolerated
- It is interpreted as a traditional regularization parameter that can be estimated by Cross Validation, for example

Non-linear data

- SVM allow for a computationally efficient method of transforming the dataset to higher dimensions using *kernel trick*.
- Common kernels that are used are
  - Linear, polynomial, Gaussian RBF, Sigmoid

- linear: $\langle x, x' \rangle$.
- polynomial: $(\gamma \langle x, x' \rangle + r)^d$, where d is specified by parameter `degree`, r by `coef0`.
- rbf: $\exp(-\gamma \|x - x'\|^2)$, where γ is specified by parameter `gamma`, must be greater than 0.
- sigmoid $\tanh(\gamma \langle x, x' \rangle + r)$, where r is specified by `coef0`.