cct | College Dublin
Computing • IT • Business

Machine Learning for Data Analysis
MSc in Data Analytics
CCT College Dublin

Linear Models for Regression & Classification
Week 3

Lecturer: Dr. Muhammad Iqbal*

Email: miqbal@cct.ie

Agenda

cct | College Dublin
Computing • IT • Business

- Introduction to Regression
- Regression Analysis
- Linear Regression
- Ridge Regression (**L_2 Regularization**)
- Lasso Regression (**L_1 Regularization**)
- Comparison of Linear, Ridge and Lasso Regression
- Linear Regression for Multi-class classification
- Linear and Logistic Regression
- Strengths and Weaknesses

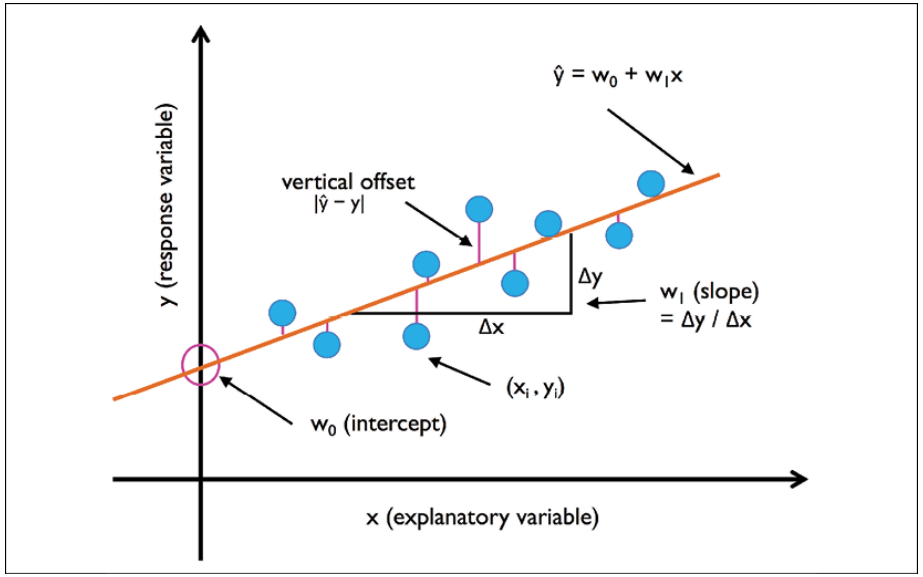
Introduction to Regression

cct | College Dublin
Computing • IT • Business

- Regression models are used to predict target variables on a continuous scale, which makes them attractive for addressing many questions in various fields.
- They have applications in industry, such as understanding relationships between variables, evaluating trends, or making forecasts. One example is predicting the sales of a company in the future months.
- **Example:** Estimate a patient's systolic blood pressure, based on patient's age, gender, body-mass index, and sodium levels.

  a) Use training data to develop the model that estimates systolic blood pressure based on predictor variables.

  b) Apply model to new cases in order to obtain estimated systolic blood pressure.



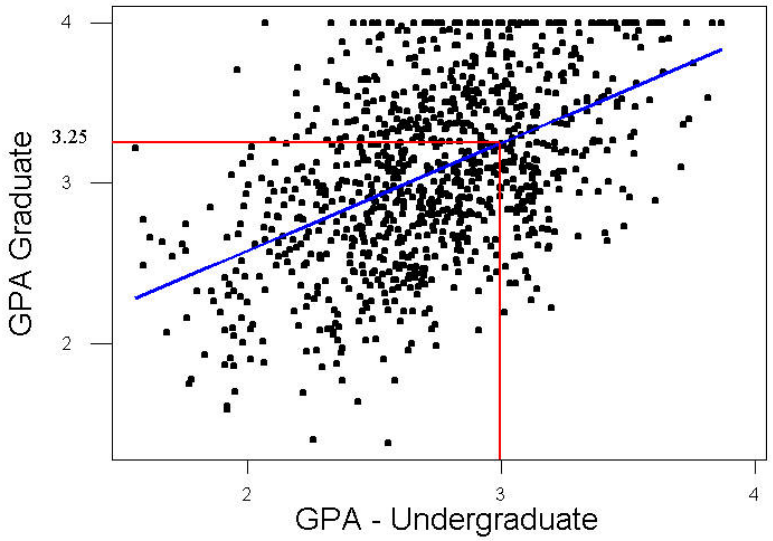
Supervised Learning Use Cases Examples

Domain	Question
Retail	How much will be the daily, monthly, and yearly sales for a given store for the next 3 years?
Retail	How many car park spaces should be allocated to a retail store?
Manufacturing	How much will be the productwise manufacturing labor cost?
Manufacturing / Retail	How much will be my monthly electricity cost for the next 3 years?
Banking	What is the credit score of a customer?
Insurance	How many customers will claim the insurance this year?
Energy / Environmental	What will be the temperature for the next 5 days?

Introduction to Regression

cct College Dublin
Computing • IT • Business

- Figure shows scatter plot of graduate **GPA** against undergraduate **GPA** (1000 students).
- **Linear regression** finds line (blue) best approximating relationship between two variables.
- **Regression line** estimates student's graduate **GPA** based on their undergraduate **GPA**, resulting in the following model as $\mathbf{w_0 = 1.24}$ and $\mathbf{w_1 = 0.67}$.
- $\mathbf{\hat{y} = w_0 + w_1 x \quad => \quad \hat{y} = 1.24 + 0.67x}$
- For example, suppose student's undergraduate **GPA = 3.0**
- According to estimation model, the estimated student's graduate **GPA** is
- **GPA = 1.24 + 0.67(3.0) = 3.25**
- Point (x = 3.0, ŷ = 3.25) lies on regression line



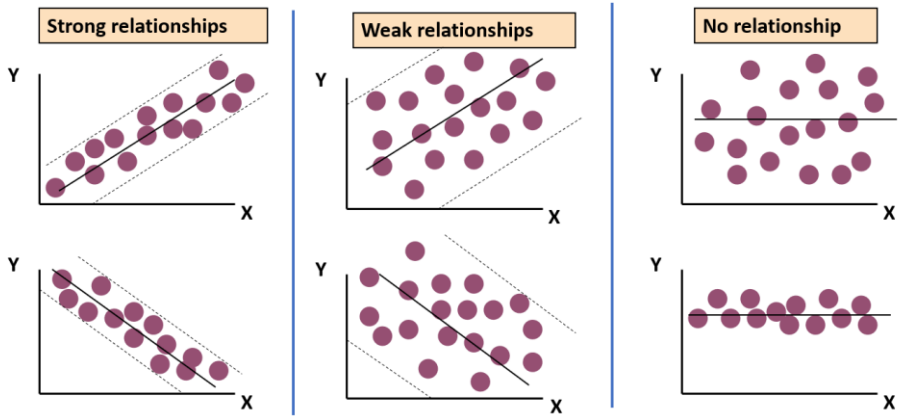
- Methods used for classification and prediction applicable to prediction.
- Includes point estimation, confidence interval estimation, linear regression and correlation, multiple regression, k-nearest neighbor, decision trees and neural networks.

Regression Analysis

cct College Dublin
Computing • IT • Business

- **Regression analysis is used to**
  - Predict the value of a **dependent variable** based on the value of at least one **independent variable.**
  - Explain the impact of changes in an independent variable on the dependent variable.
- **Dependent variable (Target variable)**
  - The variable we wish to predict or explain.
- **Independent variable (Input features)**
  - The variable used to predict or explain the dependent variable.
- The goal is to find the best $\mathbf{w_0}$ and $\mathbf{w_1}$ values, we get the best fitting of the line. So, when we are finally using our model for prediction, it will predict the value of **y** for the input value of **x**.

$$minimize \frac{1}{n} \sum_{i=1}^{n} (pred_i - y_i)^2$$



$$y = w_0 + w_1 x$$

- While training the model, we are given
  x: input training data (univariate – one input variable (parameter))
  y: labels to data (supervised learning)
- When training the model, it fits the best line to predict the value of **y** for a given value of **x**. The model gets the best regression fit line by finding the best $\mathbf{w_1}$ value.
  $\mathbf{w_0}$**:** intercept or used symbol **c**
  $\mathbf{w_1}$**:** coefficient of **x**

Linear Models

cct | College Dublin
Computing • IT • Business

- **Linear models (LMs)** are a class of models that are widely used in practice and LMs make a prediction using a ***linear function*** of the input features.

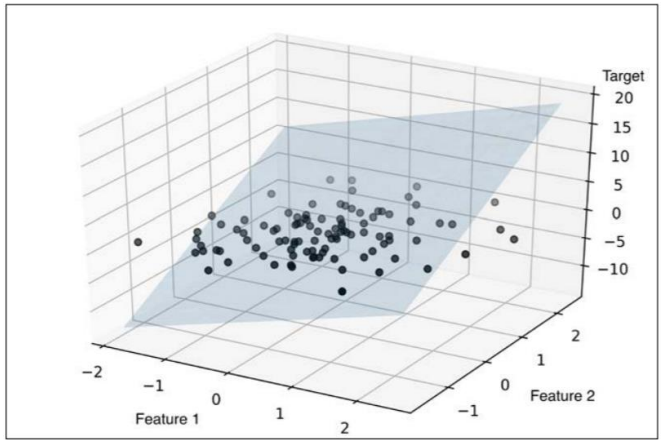
- For multiple regression, the general prediction formula for a linear model is mentioned below

$$\hat{y} = w_0 + w_1x_1 + w_2x_2 + w_3x_3 + \ldots + w_nx_n$$

- where x_1 to x_m shows the features (for example, columns/ attributes in a file) of a single data point, $w[1\ldots n]$ and w_0 are parameters of the model that are learned, and $\hat{y}$ is the prediction the model makes. For a dataset with a single feature, we can mention as

  - $\hat{y} = w[0] + w[1] * x[1]$

- $w[1]$ is the slope and $w[0]$ is the y-axis offset (y - intercept). For more features, w contains the slopes along each feature axis. Alternatively, we can think of the predicted response as being a weighted sum of the input features, with weights (which can be negative) given by the entries of w.



- Visualizing multiple linear regression fits in 3-dimensional scatter plot.

- **Multiple regression** is an extension of linear regression models that allow predictions of systems with **multiple independent variables**.

Linear Regression

aka Ordinary Least Squares

cct | College Dublin
Computing • IT • Business

- **Linear regression**, or ***ordinary least squares*** **(OLS)**, is the simplest and most classic linear method for Regression.
- **Linear regression** finds the parameters ***w*** and ***c*** that minimize the ***mean squared error*** between predictions and the true regression targets, ***y***, on the training set.
- **R^2 (coefficient of determination) Regression Score Function**
- Best possible score is 1.0 and it can be negative (because the model can be arbitrarily worse). A constant model that always predicts the expected value of y, disregarding the input features, would get a R^2 score of 0.0.
- For example, ***R^2*** of around 0.66 is not very good, but we can see that the scores on the training and test sets are very close together. This means we are likely **underfitting**, not **overfitting**.
- For this one-dimensional dataset, there are less chances of **overfitting**, as the model is very simple (or restricted). With **higher-dimensional datasets** (meaning datasets with a large number of features), linear models become more powerful, and there is a higher chance of overfitting.

Ridge Regression

L_2 Regularization

cct College Dublin
Computing • IT • Business

- **Regularization** means explicitly restricting a model to avoid overfitting. In order to create less complex model when we have a large number of features in the dataset, some of the Regularization techniques used to address over-fitting and feature selection from the dataset.
- **Ridge regression** is a linear model for regression based on similar formula as we used for the linear regression (ordinary least squares). In ridge regression, the **coefficients (*w*)** are used for training the data, but also to fit an additional constraint.
- L_2 regularization forces the weights to be small but does not make them zero and does non sparse solution. L_2 tends to shrink the coefficients evenly. The particular kind used by Ridge Regression is known as L_2 regularization.
- **Ridge regression** adds "squared magnitude" of coefficient as penalty term to the loss function. It is clear from the highlighted part represents L_2 regularization element.

Cost function of Ridge Regression

$$\sum_{i=1}^{n}(y_i - \sum_{j=1}^{p} x_{ij}\beta_j)^2 + \alpha\sum_{j=1}^{p}\beta_j^2$$

Comparison

Linear and Ridge Regression

cct College Dublin
Computing • IT • Business

- If we obtain the lower training set score using Ridge than **LinearRegression**, while the test set score is *higher*.
- Ridge is a more restricted model and there are less chances of overfitting. A less complex model means worse performance on the training set, but better generalization.
- We are always interested in generalization performance; we should choose the Ridge model over the **LinearRegression** model.
- The **Ridge model** makes a trade-off between the simplicity of the model (near-zero coefficients) and its performance on the training set.
- How much importance the model places on simplicity versus training set performance can be specified by the user, using the **alpha (α)** parameter.

L1 Regularization

$$\text{Cost} = \sum_{i=0}^{N} (y_i - \sum_{j=0}^{M} x_{ij}\,\beta_j)^2 + \alpha \sum_{j=0}^{M} |\beta_j|$$

L2 Regularization

$$\text{Cost} = \underbrace{\sum_{i=0}^{N} (y_i - \sum_{j=0}^{M} x_{ij}\,\beta_j)^2}_{\text{Loss function}} + \underbrace{\alpha \sum_{j=0}^{M} \beta_j^2}_{\text{Regularization Term}}$$

L_1 & L_2 maths. (Valverde, 2018)

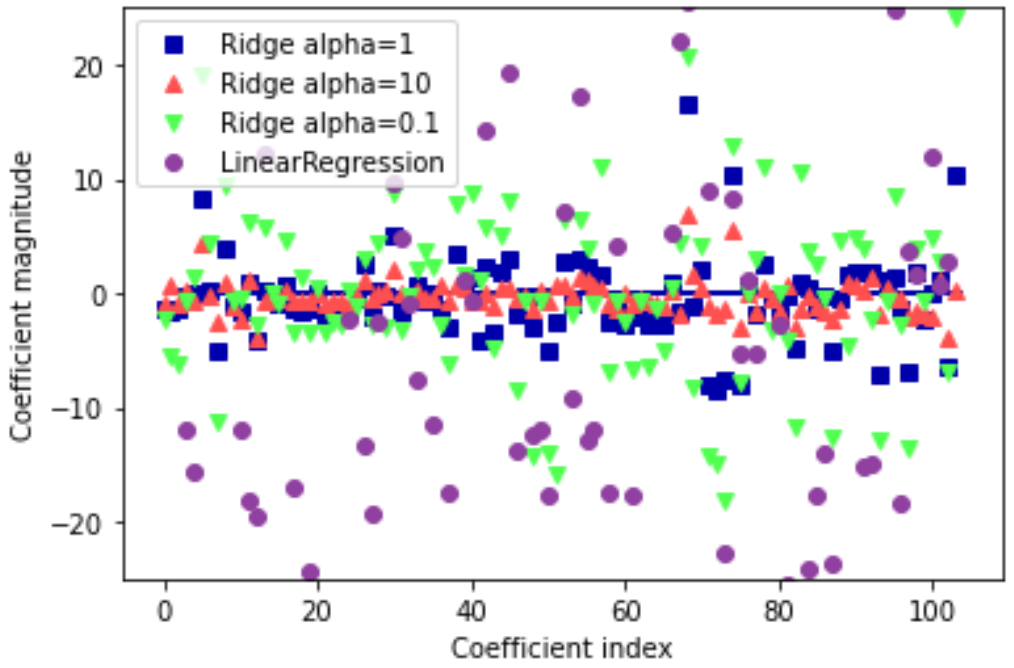
Valverde, J. M., 2018. *Lipman's Artificial Intelligence Directory.*

Ridge Regression

Effect of alpha (α)

$$\sum_{i=1}^{n}(y_i - \sum_{j=1}^{p} x_{ij}\beta_j)^2 + \alpha \sum_{j=1}^{p} \beta_j^2$$

- We can obtain best trade-off by tuning the **alpha (α)** parameter.
- The optimum setting of **alpha (α)** depends on the particular dataset that we are using.
- We can get a more qualitative insight into how the **alpha (α)** parameter changes the model by inspecting the **coef_** attribute of models with different values of **alpha (α)**.
- A higher **alpha (α)** means a more restricted model, so we can expect the entries of **coef_** to have smaller magnitude for a high value of **alpha (α)** than for a low value of **alpha (α)**.



*Comparing coefficient magnitudes for ridge regression with **different values of alpha** and linear regression*

Lasso Regression

L_1 Regularization

cct College Dublin
Computing • IT • Business

- **Lasso** can be used as a substitute for Ridge when regularizing linear regression. Using the Lasso restricts coefficients in Ridge Regression to be near to zero in a similar but somewhat different manner known as **L1 regularization**.
- The consequence of **L_1 regularization** is that when using the Lasso, some coefficients are *exactly zero*.
- This means some features are entirely ignored by the model. Having some coefficients be exactly zero makes a model easier to interpret, and can reveal the most important features of your model.
- **Lasso Regression** (Least Absolute Shrinkage and Selection Operator) adds "absolute value of magnitude" of coefficient as penalty term to the loss function. Let's apply the Lasso to the extended Boston Housing dataset:

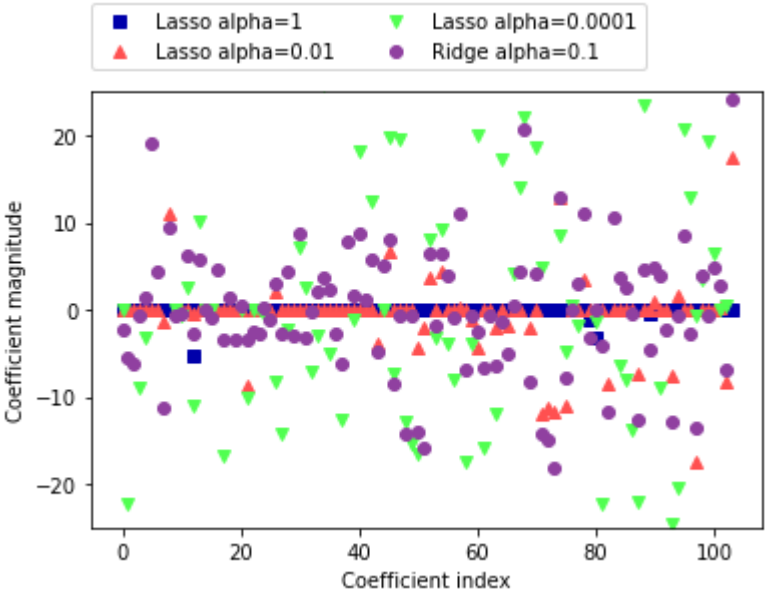
Cost function of Lasso Regression $\sum_{i=1}^{n}(Y_i - \sum_{j=1}^{p} X_{ij}\beta_j)^2 + \alpha\sum_{j=1}^{p}|\beta_j|$

Comparison of CM

Lasso Regression

- **For alpha (α) = 1**, most of the coefficients are zero, but that the remaining coefficients are also small in magnitude.
- **For alpha (α) = 0.01**, we obtain the solution shown as the red dots, which causes most features to be exactly zero.
- **For alpha (α) = 0.00001**, we get a model that is quite unregularized, with most coefficients nonzero and of large magnitude as shown with green dots.
- For comparison, the best **Ridge solution** is shown in teal.
- The Ridge model with alpha (α) = 0.1 has similar predictive performance as the Lasso model with alpha (α) = 0.01, but using Ridge, all coefficients are nonzero.

CM: Coefficient magnitude



Comparing coefficient magnitudes for Lasso regression with different values of alpha and ridge regression

Comparison of CM

Lasso Regression

cct | College Dublin
Computing • IT • Business

- **Ridge Regression** is the first choice between these two models.
- **Lasso** would be a better option if we have a lot of features but only think a few of them will be crucial.
- Similar to this, Lasso will offer a model that is simpler to comprehend and will only choose a portion of the input features if we want a model that is simple to read.
- **scikit-learn** provides the **ElasticNet** class, which combines the penalties of **Lasso** and **Ridge**.
- In practice, sometimes this combination works better, though at the price of having two parameters to adjust: one for the L_1 **regularization**, and one for the L_2 **regularization**.
- Linear regression with combined L_1 and L_2 priors as regularizer.

```python
from sklearn.linear_model import ElasticNet

# Initialise an object (e_bet) by calling a method ElasticNet()
e_net = ElasticNet(alpha = 1)

# Train the model by calling a fit() method
e_net.fit(X_train, y_train)

# Calculate the prediction and mean square error
y_pred_elastic = e_net.predict(X_test)

# Calculate the mean square error
mean_squared_error = np.mean((y_pred_elastic - y_test)**2)

# Display the mean square error
print("Mean Squared Error on test set", mean_squared_error)
```

```
Mean Squared Error on test set 63.58630803113862
```

- **alpha (α)** value between 0 - 1 will perform an elastic net.
- When **alpha (α) = 0.5** we perform an equal combination of penalties whereas **alpha (α) < 0.5** will have a heavier ridge penalty applied and **alpha (α) > 0.5** will have a heavier Lasso penalty.

Linear Models

Classification

cct College Dublin
Computing • IT • Business

- **Linear models** are extensively used for classification. Let's look at binary classification first.
- In this case, a **prediction** is made using the following formula
  - $f(x) = \hat{y} = w[0] * x[0] + w[1] * x[1] + \ldots + w[p] * x[p] + c > 0$
- The formula is fairly similar to that for linear regression, however we cutoff the projected value at zero rather than returning the weighted sum of the characteristics.
- If the **function** (***f*(x)**) is smaller than zero, we predict the class as **-1**; if it is larger than zero, we predict the class **+1**. This prediction rule is common to all linear models for classification.
- So there are many different ways to find the **coefficients (*w*)** and the **intercept (c)**.

Linear Models

Classification

cct College Dublin
Computing • IT • Business

- For linear models of regression, the output, $\hat{y}$, is a linear function of the features: **a line**, **plane**, or **hyperplane** (in higher dimensions).

- For linear models of classification, the ***decision boundary*** is a linear function of the input.

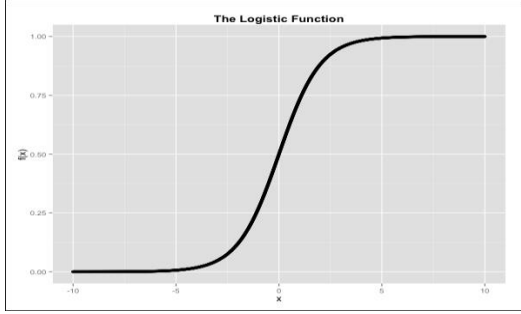
- There are many algorithms for learning linear models. These algorithms differ in the following two ways

  - The way in which they measure how well a particular **combination of coefficients** and **intercept** fits the training data

  - If and what kind of **regularization** they use

- Different algorithms choose different ways to measure what "fitting the training set well" means.

Logistic Regression



- A supervised method that works for classification is logistic regression. Using the **Logistic Sigmoid Function (*f*(*x*))**, Logistic Regression, which is based on linear regression, changes its output and produces a probability value that maps various classes.

The form of the logistic (Sigmoid) function is as follows

$$f(x) = \frac{e^x}{e^x + 1} = \frac{e^{-x}}{e^{-x}} \cdot \frac{e^x}{\left(e^x + 1\right)} = \frac{1}{1 + e^{-x}}$$

- In **Logistic Regression**, the input features are linearly scaled as with linear regression.

- The logistic function receives the outcome as an argument. This function applies a nonlinear transformation to the input and makes sure that the output's range, which may be seen as the likelihood that the input belongs to class **1**, is within the range **[0, 1]**.

$$P(y_i = 1 \mid X) = \frac{1}{1 + e^{-(w_0 + w_1 x)}}$$

- A Logistic regression is a widely used binary classifier (i.e., the target vector can only take two values). In the Logistic regression, a linear model (e.g., $w_0 + w_1 x$) is included in the Logistic (also called sigmoid) function.

- The effect of the logistic function is to constrain the value of the function's output to between **0** and **1** so that it can be interpreted as the probability. If $P(y_i = 1 \mid X)$ is greater than **0.5**, class **1** is predicted; otherwise, class **0** is predicted.

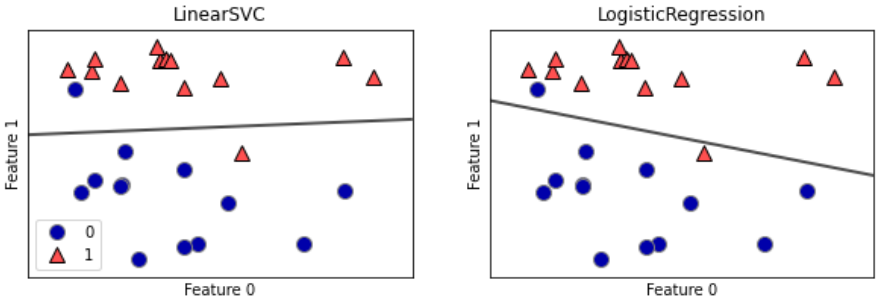
Linear and Logistic Regression

- The two most common linear classification algorithms are ***Logistic regression***, implemented in **linear_model.LogisticRegression**, and ***Linear support vector machines*** (linear SVMs), implemented in **svm.LinearSVC** (SVC stands for support vector classifier).

- In the case of Linear Regression, the outcome is continuous while in the case of Logistic Regression, the outcome is discrete (not continuous) or categorical.

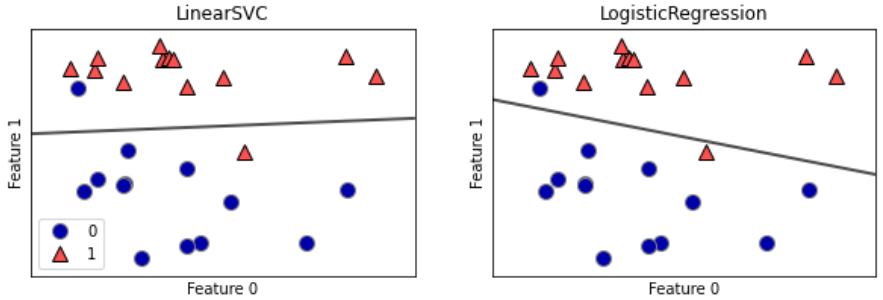
- We can apply the **LogisticRegression** and **LinearSVC** models to the forge dataset, and visualize the decision boundary as found by the linear models.

```
from sklearn.linear_model import LogisticRegression
from sklearn.svm import LinearSVC

X, y = mglearn.datasets.make_forge()

fig, axes = plt.subplots(1, 2, figsize=(10, 3))

for model, ax in zip([LinearSVC(), LogisticRegression()], axes):
    clf = model.fit(X, y)
    mglearn.plots.plot_2d_separator(clf, X, fill=False, eps=0.5, ax=ax, alpha=.7
    mglearn.discrete_scatter(X[:, 0], X[:, 1], y, ax=ax)
```



Decision boundaries of a **Linear SVM** and **Logistic regression** on the forge dataset with the default parameters

Linear and Logistic Regression



- We can see the decision boundaries found by **LinearSVC** and **LogisticRegression** respectively as straight lines, separating the area classified as **class 1** on the top from the area classified as **class 0** on the bottom (**Feature 0** along x-axis and **Feature 1** along y-axis)

- Any new data point that lies above the black line will be classified as **class 1** by the respective classifier, while any point that lies below the black line will be classified as **class 0**.

- For **LogisticRegression** and **LinearSVC,** the trade-off parameter that determines the strength of the regularization is called **C**, and higher values of **C** correspond to *less* regularization.

- If we use a high value of **C**, **LogisticRegression** and **LinearSVC** try to fit the training set as best as possible, while with low values of the parameter **C**, the models put more emphasis on finding a **coefficient vector (*w*)** that is close to zero.

Linear Models

Multiclass Classification

cct College Dublin
Computing • IT • Business

- Many linear classification models are for binary classification only, and we need to extend to the multiclass case. A common technique to extend a binary classification algorithm to a multiclass classification algorithm is the ***one-vs.-rest* approach**.

- In the **one-vs.-rest approach**, a binary model is learned for each class that tries to separate that class from all the other classes, resulting in as many binary models as there are classes.

- To make a prediction, all **binary classifiers** are run on a test point. The classifier that has the highest score on its single class "wins," and this class label is returned as the **prediction**.

- Having one binary classifier per class results in having **one vector of coefficients (*w*)** and **one intercept (*c*)** for each class. The class for which the result of the classification confidence formula given is highest is the assigned class label

  - $w[0] * x[0] + w[1] * x[1] + \ldots + w[p] * x[p] + c$

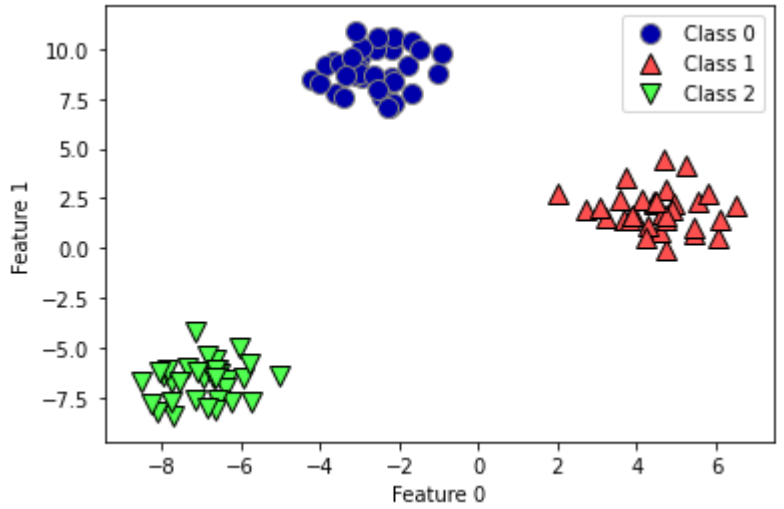
Linear Models

Multiclass Classification

cct College Dublin
Computing • IT • Business

- Let us apply the one-vs.-rest method to a simple three-class classification dataset.
- We use a two-dimensional dataset, where each class is given by data sampled.
- It is clear from the figure that the Two-dimensional toy dataset containing three classes.

```
from sklearn.datasets import make_blobs

X, y = make_blobs(random_state=42)
mglearn.discrete_scatter(X[:, 0], X[:, 1], y)
plt.xlabel("Feature 0")
plt.ylabel("Feature 1")
plt.legend(["Class 0", "Class 1", "Class 2"])
```

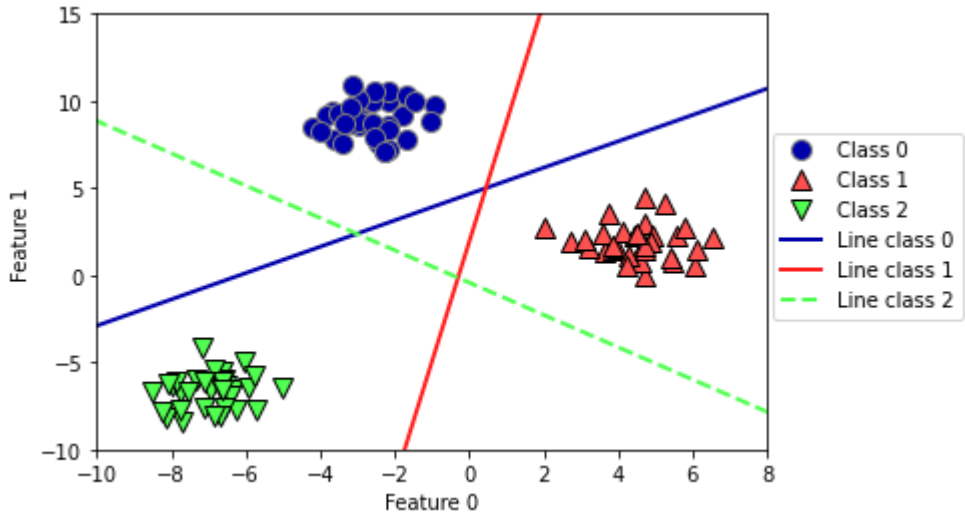


Two-dimensional Toy Dataset

Three Classes

- We can see that all the points belonging to **class 0** in the training data are above the line corresponding to **class 0**, which means that they are on the "**class 0**" side of this binary classifier.
- The points in **class 0** are above the line corresponding to **class 2**, which means they are classified as "**rest**" by the binary classifier for **class 2**.
- The points belonging to **class 0** are to the left of the line corresponding to **class 1**, which means the binary classifier for **class 1** also classifies them as "**rest**."
- Therefore, any point in this area will be classified as **class 0** by the final.
- How about the triangle in the plot's center? There, points are classified as "rest" by all three binary classifiers. **Which class would be given that point?** The class of the closest line, which has the highest value in the classification algorithm, is the correct response.

```
mglearn.discrete_scatter(X[:, 0], X[:, 1], y)
line = np.linspace(-15, 15)
for coef, intercept, color in zip(linear_svm.coef_, linear_svm.intercept_,
                                  mglearn.cm3.colors):
    plt.plot(line, -(line * coef[0] + intercept) / coef[1], c=color)
plt.ylim(-10, 15)
plt.xlim(-10, 8)
plt.xlabel("Feature 0")
plt.ylabel("Feature 1")
plt.legend(['Class 0', 'Class 1', 'Class 2', 'Line class 0', 'Line class 1',
            'Line class 2'], loc=(1.01, 0.3))
```



Decision boundaries learned by the three one-vs.-rest classifiers

Measuring The Quality Of
A Regression Model

- A useful statistics is (**s**), the standard error of the estimate is calculated as

$$s = \sqrt{\text{MSE}}$$

$$MSE = \frac{1}{n}\Sigma\underbrace{\left(y - \hat{y}\right)^2}_{\text{The square of the difference between actual and predicted}}$$

- where **SSE** is the sum of squared errors. The statistic **s** is useful for assessing the quality of a regression model because its value indicates a measure of the size of the "typical" prediction error.

$$SST = \sum_{i=1}^{n}(y_i - \bar{y})^2$$

- Another measure of the quality of a regression model is the r^2 (r-squared) statistic. r^2 measures how closely the linear regression fits the data, with values closer to 100% indicating a more perfect fit. The formula for r^2 is

$$SSR = \sum_{i=1}^{n}(\hat{y}_i - \bar{y})^2$$

$$r^2 = \frac{\text{SSR}}{\text{SST}}$$

$$SSE = \sum_{i=1}^{n}(y_i - \hat{y}_i)^2$$

- Where **SST** represents the variability in the y-variable, and **SSR** represents the improvement in estimation from using the regression model as compared to just using $\bar{y}$ to estimate **y**. Thus, r^2 may be interpreted as the ratio of the total variability in y that is accounted for by the linear relationship between ***x*** and ***y***.

Strengths and Weaknesses

cct College Dublin
Computing • IT • Business

- **Linear models** are very fast to train, and also fast to predict. They scale to very large datasets and work well with sparse data. This is true if the dataset has highly correlated features; in these cases, the coefficients might be hard to interpret.
- Linear models perform well when the number of features is large compared to the number of samples.
- They are applied to very huge datasets for the simple reason that other models cannot be trained. Other models might perform more generally in lower-dimensional spaces.
- Large values for ***alpha*** or small values for **C** mean simple models. For the regression models, tuning these parameters is quite important.
- The other decision we have to make is whether we want to use $\mathbf{L_1}$ **regularization** or $\mathbf{L_2}$ **regularization**. If we assume that only a few of features are important, we should use $\mathbf{L_1}$.
- The main parameter of linear models is the **Regularization Parameter**, called ***alpha*** in the regression models and **C** in **LinearSVC** and **LogisticRegression**.

Resources/ References

- Mastering Machine Learning with Python in Six Steps: A Practical Implementation Guide to Predictive Data Analytics Using Python Manohar Swamynathan, 1484249461, 2019, Apress.
- Introduction to Machine Learning with Python, Andreas C. Müller and Sarah Guido, O'Reilly Publishers, 2017.
- Data Mining And Machine Learning, Fundamental Concepts And Algorithms, MOHAMMED J. Zaki, Wagner Meira, Jr., Cambridge CB2 8BS, United Kingdom, 2020.
- Discovering Knowledge In Data: An Introduction To Data Exploration, Second Edition, By Daniel Larose And Chantal Larose, John Wiley And Sons, Inc., 2014.
- www.datacamp.com
- www.analyticsvidhya.com/blog/2018/08/k-nearest-neighbor-introduction-regression-python
- Some images are used from google search repository to enhance the level of learning.
- Sample datasets for Regression: www.kaggle.com/tags/regression