Supervised Learning

Supervised Learning

- Types of Supervised Machine Learning Algorithms
  - Classification
  - Regression
- **Classification**
  - The goal is to predict a class label from a predefined list of possibilities.
  - **Binary Classification**
    - distinguishing between exactly two classes
  - **Multi-Class Classification**
    - distinguishing between more than two classes
  - **Multi Label Classification**
    - Multi Label classification is also known as multi-output classification are variants of the classification problem where multiple labels may be assigned to each instance.

Classification

- In binary classification we often speak of one class being the positive class and the other class being the negative class.
- Here, positive doesn't represent having benefit or value, but rather what the object of the study is. So, when looking for spam, "positive" could mean the spam class.
- Which of the two classes is called positive is often a subjective matter, and specific to the domain.

Regression

- The goal is to predict a **continuous number**, or a **floating point number** in programming terms (a real number in mathematical terms).
    - Ex: Predicting a person's annual income from their education, their age and where they live, is an example of a regression task.

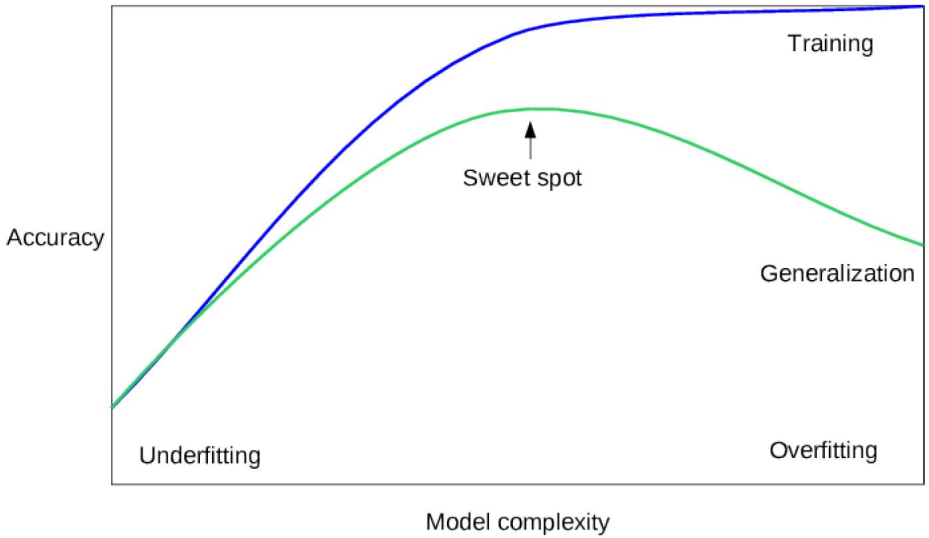
How to determine whether a given is a Classification or Regression task?

- An easy way to distinguish between classification and regression tasks is to ask whether there is some kind of ordering or continuity in the output.
- If there is an ordering, or a continuity between possible outcomes, then the problem is a regression problem.
    - Ex: There is a clear ordering of “making more money” or “making less money”. There is a natural understanding that $40,000 per year is between $50,000 per year and $30,000 per year. There is also a continuity in the output. Whether a person makes 40,000$ or 40,001$

Generalization, Overfitting and Underfitting

- In supervised learning, we build a model on the training data, and then we try to make accurate predictions on new, unseen data, that has the same characteristics as the training set that we used.
- **Overfitting**: Building a complex model that does well on the training set but does not generalize to new data is known as overfitting, because we are **focusing too much on the particularities of the training data**.
- **Underfitting**: Using too simple model is called underfitting, because we don't explain the target output for the training data well enough.

Model Generalization



Model Generalization

- If we choose use a **model that is too simple**, we will do **badly on the training set**, and similarly **badly on the test set**, as we would using only the mean prediction.
- The more complex we allow our model to be, the better we will be able to predict on the training data. However, if our **model becomes too complex**, we start **focusing too much on the particularities of our training set**, and the **model will not generalize well to new data.**
- There is a sweet spot in between, which will yield the best generalization performance. That is the model we want to find.

Supervised Machine Learning Algorithms

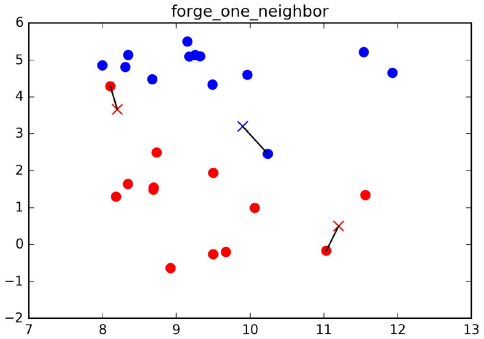
- What are we going to cover?
  - Discuss strength and weaknesses of each algorithm
  - What kind of data they can be best applied to
  - Explain the meaning of the most important parameters and options.
  - Describe the classification and a regression variant of algorithms (if available).
  - Use several datasets to illustrate the different algorithms. Some of the datasets will be small synthetic (meaning made-up) datasets, designed to highlight particular aspects of the algorithms. Other datasets will be larger, real world examples datasets.
    - Example: An example of a synthetic two-class classification dataset is the forge dataset, which has two features.

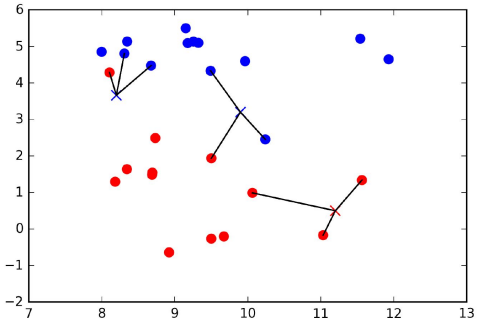
k-Nearest Neighbor

- The k-Nearest Neighbors (kNN) algorithm is arguably the simplest machine learning algorithm.
- Building the model only consists of storing the training dataset.
- To make a prediction for a new data point, the algorithm finds the closest data points in the training dataset, it “nearest neighbors”.

k-Neighbors Classification

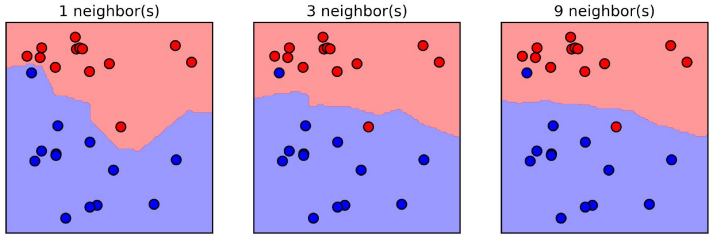
- In its simplest version, the algorithm only considers exactly one nearest neighbor, which is the **closest training data point to the point we want to make a prediction for**.
- The prediction is then simply the known output for this training point.
- When considering more than one neighbor, we use **voting to assign a label**. This means, for each test point, we count how many neighbors are red, and how many neighbors are blue. We then assign the class that is more frequent: in other words, the majority class among the k neighbors.





Analyzing KNeighborsClassifier

- As you can see in the figure, using a single neighbor results in a decision boundary that follows the training data closely.
- Considering more and more neighbors leads to a smoother decision boundary.
- A smoother boundary corresponds to a simple model.
- In other words, using **few neighbors corresponds to high model complexity** (as shown in the right side figure) and using **many neighbors corresponds to low model complexity.**



k-Neighbors Regression

- The prediction using a single neighbor is just the target value of the nearest neighbor.
- When using **multiple nearest neighbors for regression, the prediction is the average (or mean) of the relevant neighbors.**
- We can evaluate the regression model using the score method, which for regressors returns the R^2 score.
- The R^2 score, also known as **coefficient of determination, is a measure of goodness of a prediction for a regression model, and yields a score up to 1.**
- A value of **1 corresponds to a perfect prediction**, and **a value of 0 corresponds to a constant model that just predicts the mean of the training set** responses.

Parameters of k-Nearest Neighbors

- In principal, there are two important parameters to the K-Neighbors classifier:
  - **The number of neighbors**
  - **How you measure distance between data points.**
- In practice, **using a small number of neighbors** like 3 or 5 often **works well**, but you should **certainly adjust this parameter**.
- **By default, Euclidean distance is used**, which works well in many settings.

Strengths of k-Nearest Neighbors

- It is very easy to understand
- Often gives reasonable performance without a lot of adjustments.
- Using nearest neighbors is a good baseline method to try before considering more advanced techniques.
- Building the nearest neighbors model is usually very fast, but when your training set is very large (either in number of features or in number of samples) prediction can be slow.

Weaknesses of k-Nearest Neighbors

- When using nearest neighbors, it's **important to preprocess your data.**
- Nearest neighbors often **does not perform well on dataset with very many features**, in particular **sparse datasets**, a common type of data in which **there are many features, but only few of the features are non-zero for any given data point**.
- So while the nearest neighbors algorithm is easy to understand, **it is not often used in practice, due to prediction being slow, and its inability to handle many features.**

Note : The method we discuss next has neither of these drawbacks.

Linear Models

Linear Models

- Linear models are a class of models that are widely used in practice, and have been studied extensively in the last few decades, with roots going back over a hundred years.
- Linear models are models that make a **prediction that using a linear function of the input features.**
- Linear models for regression can be characterized as regression models for which the prediction is a line for a **single feature,** a plane when using two features, or a hyperplane in higher dimensions (that is when **having more features**).

Linear Models for Regression

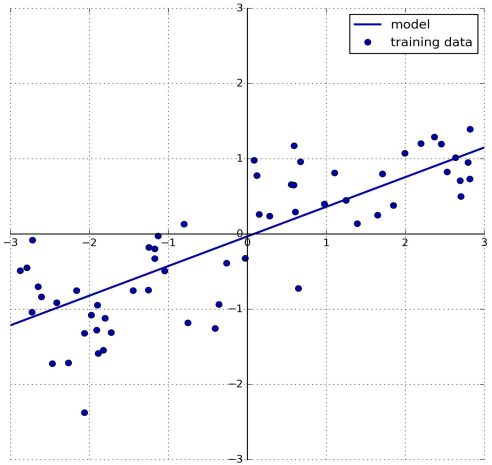
- For regression, the general prediction formula for a linear model looks as follows:

  $\hat{y} = w[0] * x[0] + w[1] * x[1] + \ldots + w[p] * x[p] + b$

  of a single data point, w and b are parameters of the model that are learned, and $\hat{y}$ is the prediction the model makes.
- For a dataset with a single feature, this is:

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

  Where w[0] is the slope and b is the y-axis offset.



Linear Models for Regression

- For datasets with many features, linear models can be very powerful. In particular, if you have more features than training data points, any target y can be perfectly modeled (on the training set) as a linear function.
- There are many different linear models for regression. The difference between these models lies in how the model parameters w and b are learned from the training data, and how model complexity can be controlled.

Linear regression (aka Ordinary Least Squares)

$$Cost\ (W) = RSS(W) = \sum_{i=1}^{N} \{y_i - \hat{y}_i\}^2 = \sum_{i=1}^{N} \left\{ y_i - \sum_{j=0}^{M} w_j\, x_{ij} \right\}^2$$

- Linear regression, or ordinary least squares (OLS), is the **simplest and most classic linear method for regression.**
- Linear regression **finds the parameters w and b that minimize the mean squared error** between predictions and the true regression targets, y, on the training set.
- Linear regression **has no parameters, which is a benefit, but it also has no way to control model complexity.**
- The "slope" parameters (w), also called weights or coefficients and (b) is offset or intercept.
- **For datasets with many features, linear models can be very powerful. In particular, if we have more features than training data points, any target y can be perfectly modeled (on the training set) as a linear function.**
- One of the **most commonly used alternatives to standard linear regression is ridge regression.**

Linear Regression - Demo

Ridge regression

$$Cost\ (W) = RSS(W) + \lambda * (sum\ of\ squares\ of\ weights)$$

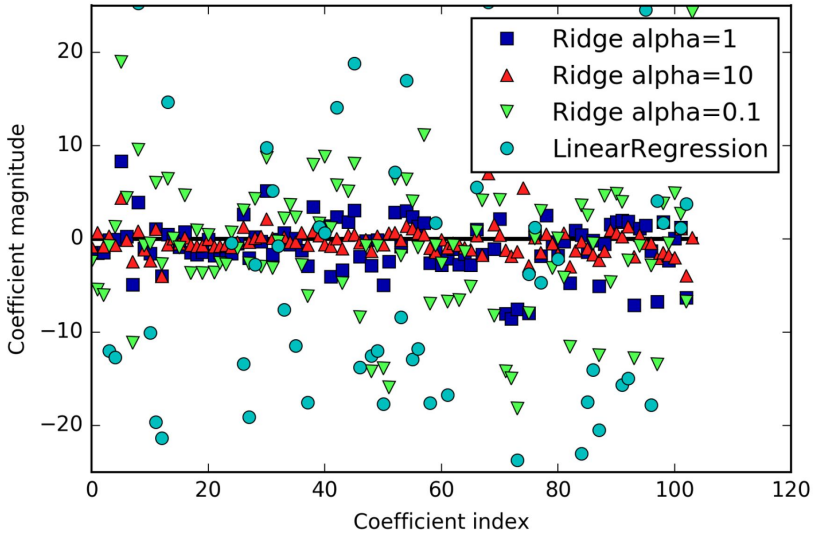
$$= \sum_{i=1}^{N} \left\{ y_i - \sum_{j=0}^{M} w_j\, x_{ij} \right\}^2 + \lambda \sum_{j=0}^{M} w_j^2$$

- Ridge regression is also a **linear model for regression**, so the formula it uses to make predictions is the same one used for ordinary least squares.
- In ridge regression, though, **the coefficients (w) are chosen not only so that they predict well on the training data, but also to fit an additional constraint.** We also want the **magnitude of coefficients to be as small as possible**; in other words, **all entries of w should be close to zero.**
- Intuitively, this means **each feature should have as little effect on the outcome as possible** (which translates to having a small slope), while still predicting well.
- This constraint is an example of what is called regularization.
- Regularization means **explicitly restricting a model to avoid overfitting**.
- **Ridge regression uses L2 Regularization to avoid overfitting.**

Ridge Regression

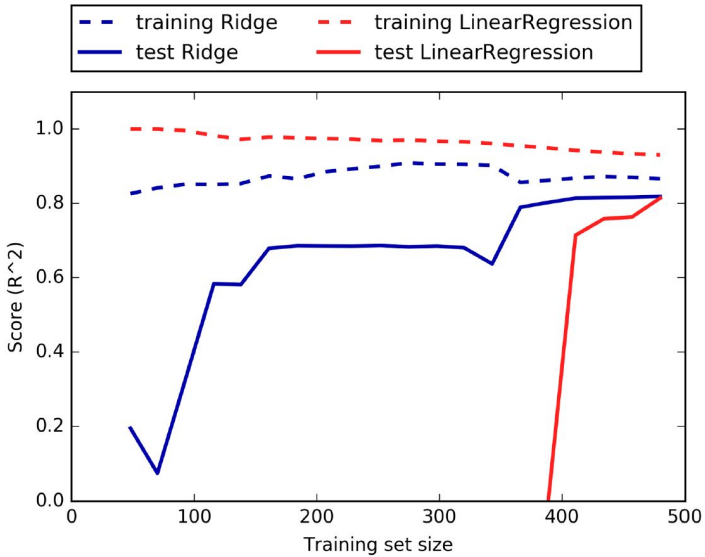
- Ridge is a more restricted model, so we are less likely to overfit.
- A less complex model means worse performance on the training set, but better generalization.
- The Ridge Regression model makes a trade-off between the simplicity of the model (near-zero coefficients) and its performance on the training set.
- How much importance the model places on simplicity versus training set performance can be specified by the user, using the alpha parameter.
- The optimum setting of alpha depends on the particular dataset we are using.
- Increasing alpha forces coefficients to move more toward zero, which decreases training set performance but might help generalization.

Impact of Regularization on Ridge Regression



- A higher alpha means a more restricted model, so we expect the entries of coefficients to have smaller magnitude for a high value of alpha than for a low value of alpha.

Impact of Regularization on Ridge Regress



Learning curves for ridge regression and linear regression on the Boston Housing dataset
Scenario: Alpha is fixed and varied the amount of training data

Key TakeAways

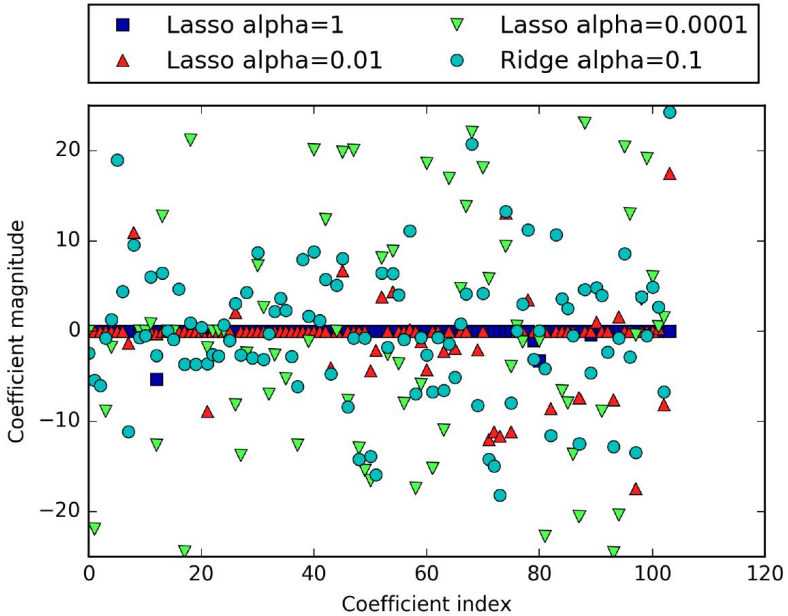
- The training score is higher than the test score for all dataset sizes, for both ridge and linear regression.
- As ridge is regularized, the training score of ridge is lower than the training score for linear regression across the board.
- The test score for ridge is better, particularly for small subsets of the data. For less than 400 data points, linear regression is not able to learn anything. As more and more data becomes available to the model, both models improve, and linear regression catches up with ridge in the end.
- The lesson here is that with enough training data, regularization becomes less important, and given enough data, ridge and linear regression will have the same performance
- From previous fig we could see that decrease in training performance for linear regression. If more data is added, it becomes harder for a model to overfit, or memorize the data.

Ridge Regression - Demo

Lasso

- Similar to ridge regression, lasso also restricts coefficients to be close to zero, but in a slightly different way, called L1 regularization.
- The consequence of L1 regularization is that when using the lasso, some coefficients are exactly zero. This means some features are entirely ignored by the model. This can be seen as a form of automatic feature selection.
- Having some coefficients be exactly zero often makes a model easier to interpret, and can reveal the most important features of your model.
- Similarly to Ridge, the Lasso also has a regularization parameter, alpha, that controls how strongly coefficients are pushed toward zero.
- The lasso penalizes the L1 norm of the coefficient vector—or in other words, the sum of the absolute values of the coefficients.
- If we set alpha too low, however, we again remove the effect of regularization and end up overfitting, with a result similar to LinearRegression.

Impact of Regularization on Lasso Regression



When to use what???

- In practice, ridge regression is usually the first choice between these two models (Rdige and Lasso)
- However, if we have a large amount of features and expect only a few of them to be important, Lasso might be a better choice.
- Similarly, if we would like to have a model that is easy to interpret, Lasso will provide a model that is easier to understand, as it will select only a subset of the input features.
- *Note: scikit-learn provides the ElasticNet class, which combines the penalties of Lasso and Ridge. In practice, this combination works best, though at the price of having two parameters to adjust: one for the L1 regularization, and one for the L2 regularization.*

Lasso Regression - Demo

Linear Models for Classification

Linear Models for Classification

- Linear models are also extensively used for classification.
- Let's look at binary classification first. In this case, a prediction is made using the following formula: $\hat{y} = w[0] * x[0] + w[1] * x[1] + \ldots + w[p] * x[p] + b > 0$
- The above formula looks very similar to the one for linear regression, but **instead of just returning the weighted sum of the features, we threshold the predicted value at zero**.
- If the function is smaller than zero, we predict the class –1; if it is larger than zero, we predict the class +1.
- This **prediction rule is common to all linear models for classification**. Again, there are many different ways to find the coefficients (w) and the intercept (b).

Linear Models for Classification

- For linear models for regression, the output, $\hat{y}$, is a linear function of the features: a line, plane, or hyperplane (in higher dimensions).
- For linear models for classification, the decision boundary is a linear function of the input. In other words, a (binary) linear classifier is a classifier that separates two classes using a line, a plane, or a hyperplane.
- There are many algorithms for learning linear models. These algorithms all 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

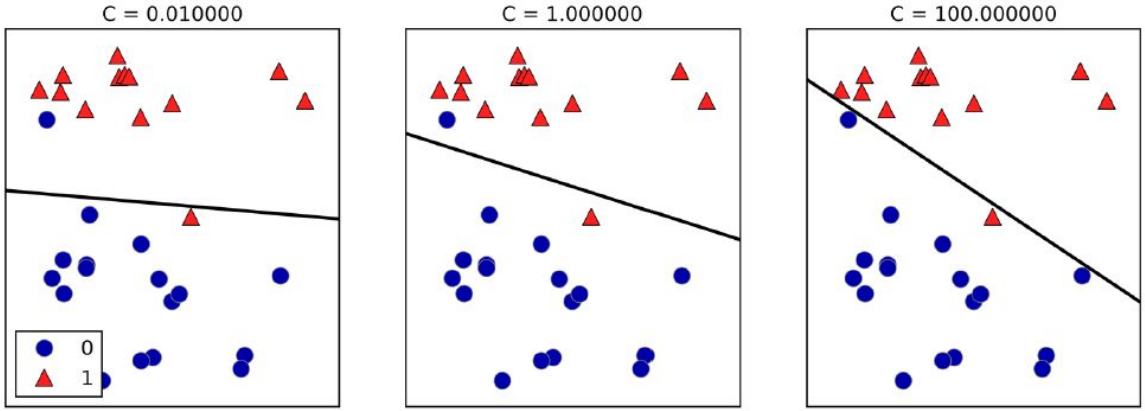
Linear Models for Classification

- Different algorithms choose different ways to measure what fitting the training set well means.
- The two most common linear classification algorithms are Logistic Regression, and Linear Support Vector Machines
- Despite its name, LogisticRegression is a classification algorithm and not a regression algorithm, and it should not be confused with LinearRegression.

Linear Models for Classification

- By default, both models **Logistic Regression and Linear Support Vector Classifier apply an L2 regularization**, in the same way that Ridge does for regression.
- 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.
- In other words, when you use a high value for the parameter 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.
- The **low values of *C* will cause the algorithms to try to adjust to the majority of data points,** while using a **higher value of *C* stresses the importance that each individual data point be classified correctly.**

Decision boundaries of a linear SVM for different values of C



- A very small C corresponding to a lot of regularization.
- For large values of C model tries hard to correctly classify all points, but might not capture the overall layout of the classes well. In other words, this model is likely overfitting.