COMP5310 — Week 4

Machine Learning

How can we learn from Data?

- Companies like AirBnB or Stayz are platforms for short-term homestays
- They would like to estimate the Customer Lifetime Value (LTV) for the listings on their platform
  - to create better listing segments,
  - and to optimize their marketing budget.

- How can they do this?
- And how can we assess the quality and accuracy of such a model?

Credit: Unsplash

Some Perspective before Getting Started

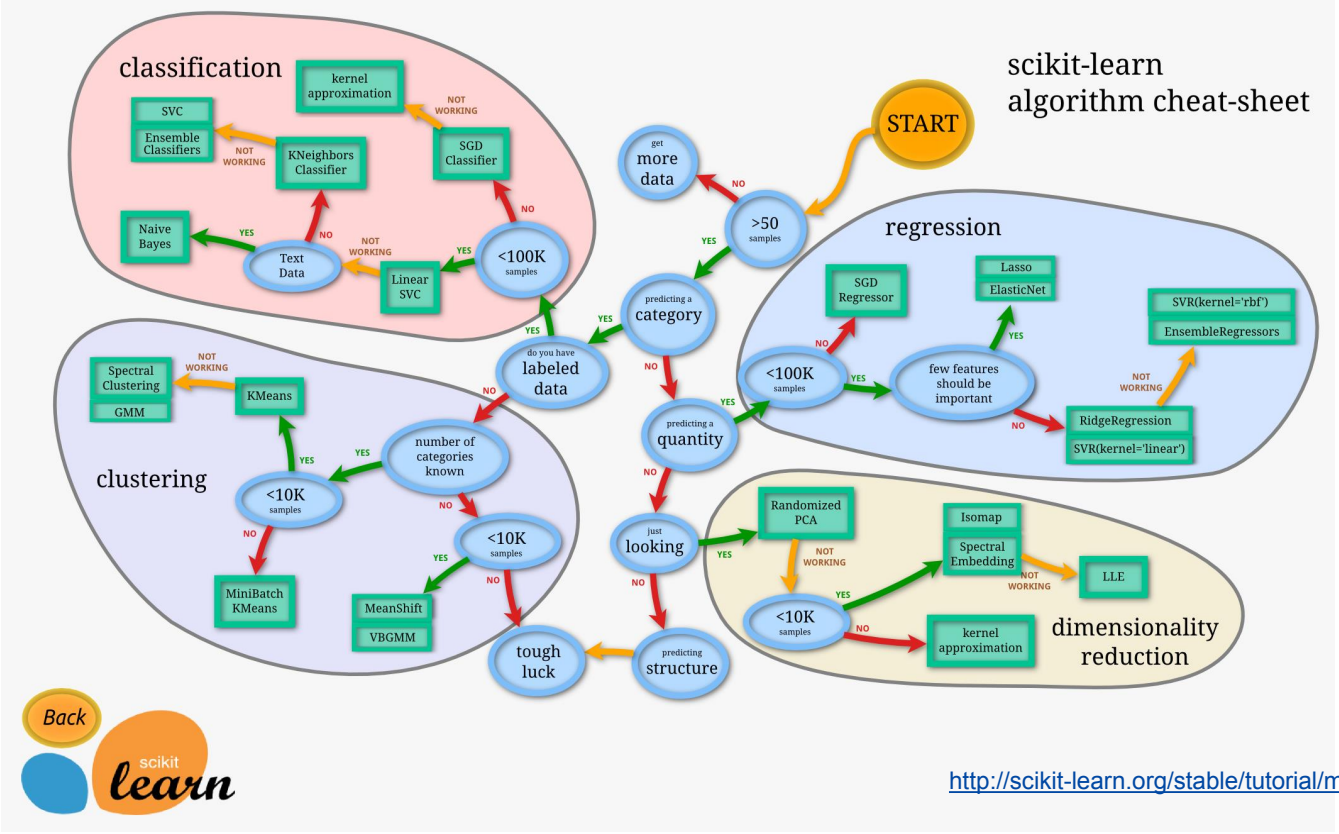
“Many people imaging that data science is mostly machine learning […] . In fact, data science is mostly turning business problems into data problems and collecting data and cleaning data and formatting data, after which machine learning is almost an afterthough.”

Data Science from Scratch, Ch. 11

What is Machine Learning?

- Creating and using models that are learned from data
  - Predicting whether an email is spam or not
  - Predicting whether a credit card transaction is fraudulent
  - Predicting which Football team will win the Champions League
- We'll focus on an *overview* of **unsupervised** and **supervised** machine learning techniques
  - Clustering
  - Simple linear regression
  - Multiple linear regression
  - Logistic regression
  - Classification / Decision trees

Machine Learning Map from scikit-learn



http://scikit-learn.org/stable/tutorial/machine_learning_map/


THE UNIVERSITY OF
SYDNEY

Unsupervised ML: Clustering

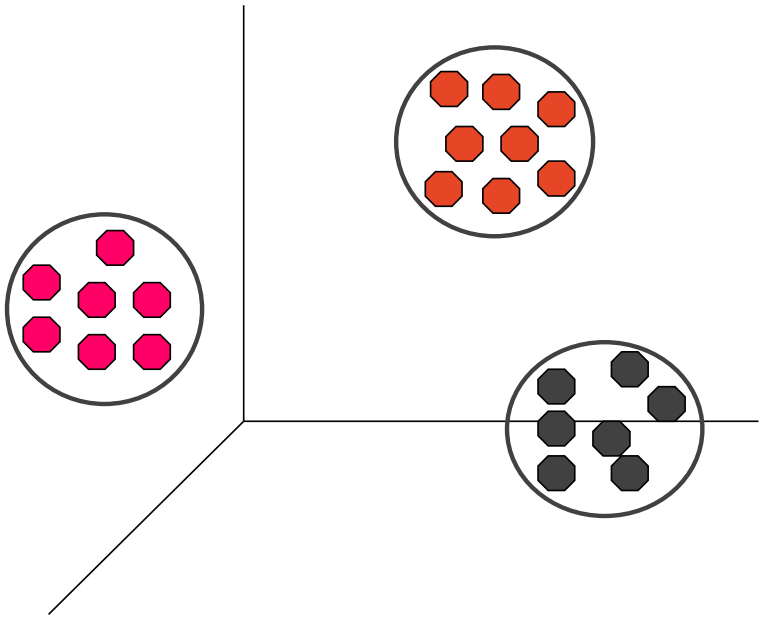
Clustering

- k-Means clustering
- DBSCAN
- Hierarchical clustering

- Given a set of *n* objects, group into *k* coherent clusters

- Distance function that specifies the "closeness" of two objects.
  ↳ Euclidean distance for example
- Fundamental problem – divide objects into cluster so that points in different clusters are far apart.

Clustering: Group Similar Objects



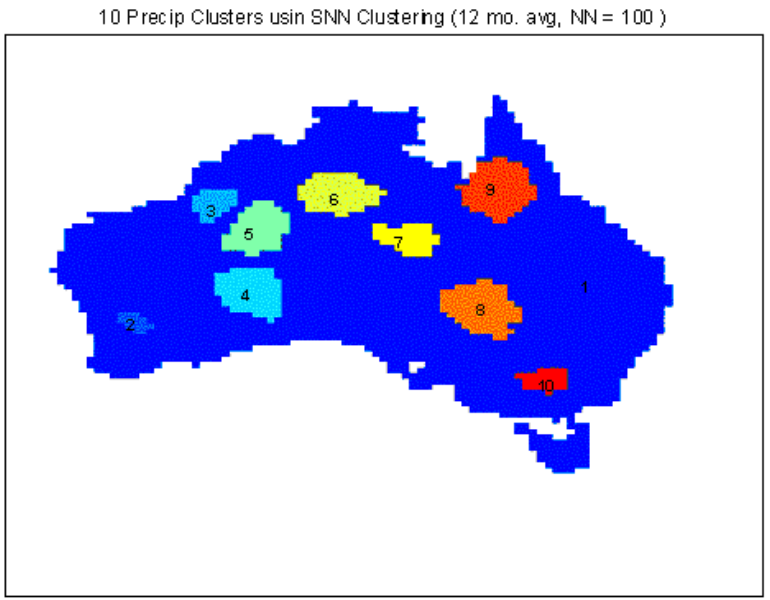
- Groups objects into clusters
- Objects in the same cluster are similar/related
- Objects in different clusters are dissimilar/unrelated

Slides adapted from Tan et al. Introduction to data mining.
http://www-users.cs.umn.edu/~kumar/dmbook/
http://www-users.cs.umn.edu/~kumar/dmbook/dmslides/chap6_basic_association_analysis.pdf

Clustering for Understanding

- Group related documents for browsing
- Group genes, proteins, or cells that have similar functionality
- Group stocks with similar price fluctuations
- etc

Clustering for Summarisation



http://www-users.cs.umn.edu/~kumar/dmbook/dmslides/chap8_basic_cluster_analysis.pdf

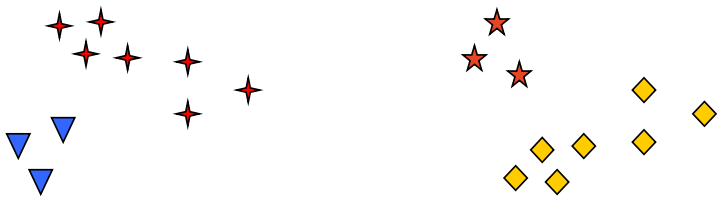
- Reduces the size of large datasets
- Summarise data before further analysis
- Vector quantisation for, e.g., images, audio, video
- etc

Notion of a Cluster can be Ambiguous

How many clusters..?

Maybe six clusters?

Two clusters?



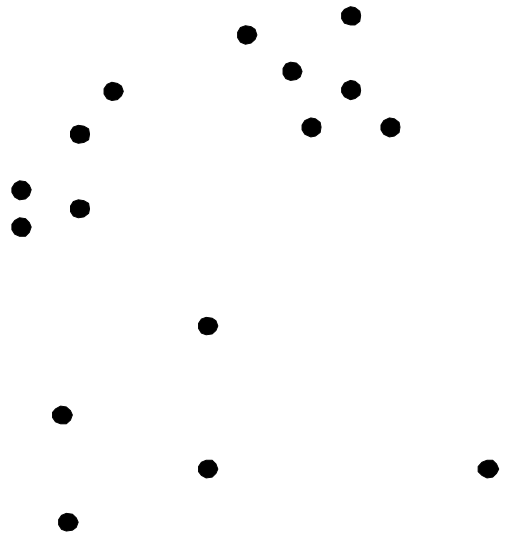
Or four clusters?

http://www-users.cs.umn.edu/~kumar/dmbook/dmslides/chap8_basic_cluster_analysis.pdf

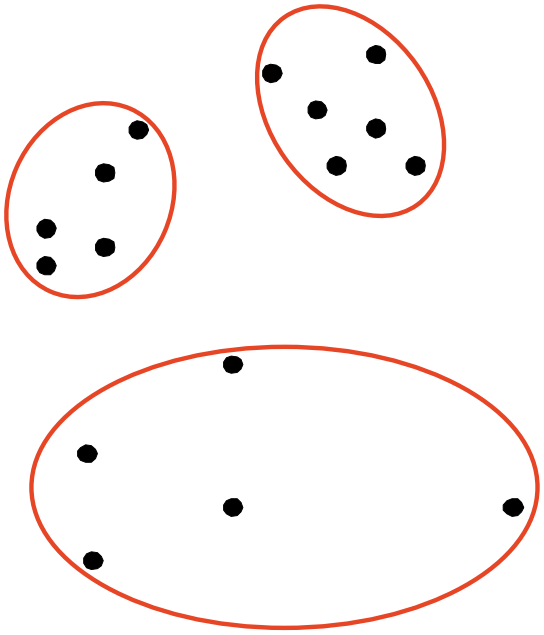
Types of Clusterings

- A **clustering** is a set of clusters
- Important distinction between hierarchical and partitional sets of clusters
- ***Partitional clustering***
  A division data objects into non-overlapping subsets (clusters) such that each data object is in exactly one subset
- ***Hierarchical clustering***
  A set of nested clusters organized as a hierarchical tree

Partitional Clustering

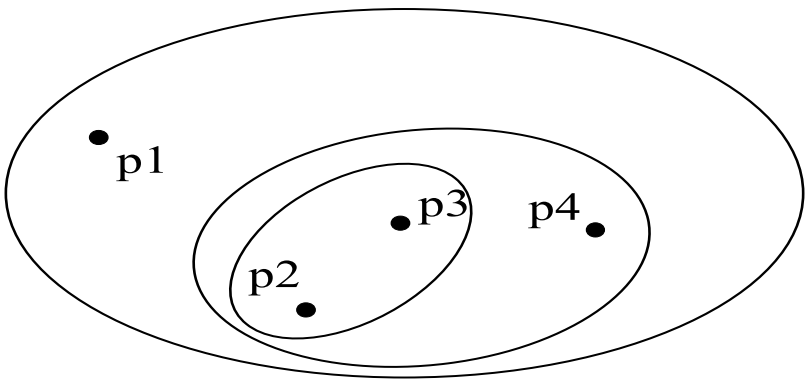


Original Points

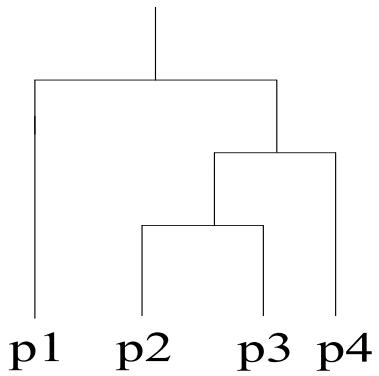


A Partitional Clustering

Hierarchical Clustering



Hierarchical clustering



Dendrogram

k-Means Clustering

- Partitional clustering approach
- Each cluster is associated with a **centroid** (center point)
- Each point is assigned to the cluster with the closest centroid
- Number of clusters, *k*, must be specified
- The basic algorithm is very simple

1: Select K points as the initial centroids.
2: **repeat**
3: Form K clusters by assigning all points to the closest centroid.
4: Recompute the centroid of each cluster.
5: **until** The centroids don't change

k-Means Clustering – Details

- Initial centroids are often chosen randomly
- Clusters produced vary from one run to another
- K-means will converge for common similarity measures
- Most of the convergence happens in the first few iterations
- Complexity is O(*n* * *k* * *i* * *d*)
  - *n* = number of points, *k* = number of clusters, *i* = number of iterations, *d* = number of attributes (or dimensions)
  - Sklearn suggests <10,000 points
  - Also need to consider *k*, *i* and *d*

depends on where the data points are located.

time complexity of K-means isn't very important to know

Overall, all of this depends on the dimensionality of the dataset.

More on Clustering

- *k*-Means clustering is a well-known, simple algorithm
  - but requires choice of *k* and is sensitive to start-choice of *k*
- Many more sophisticated algorithms available nowadays
  - eg. density-based clustering which supports unknown number of clusters
  - eg. automatic sub-space clustering (choice of relevant dimensions)
  - eg. support of per-cluster feature-selection
  - eg. outlier handling/pruning
  - etc.

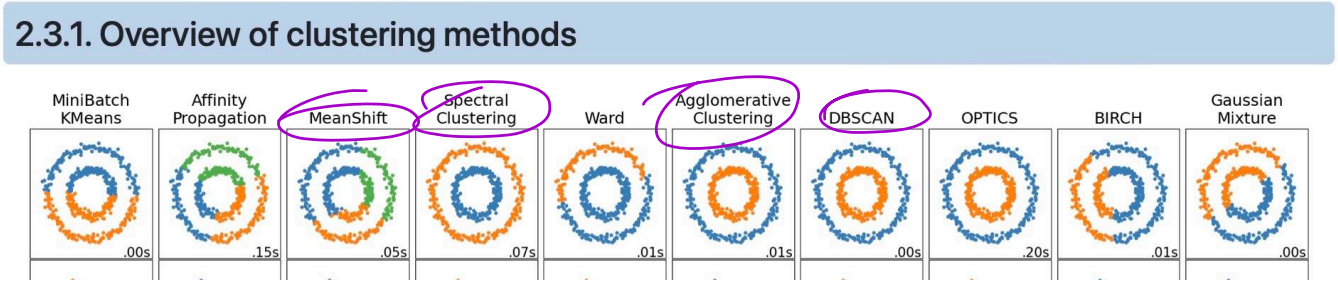
DBSCAN

Never heard of these...

- cf. Kriegel et al.: "Clustering High-Dimensional Data: A Survey on Subspace Clustering, Pattern-Based Clustering, and Correlation Clustering", ACM KDD 2009.

Scikit's sklearn.clustering

Recall DATA3001.



- k-means
- Agglomerative clustering (hierarchical clustering based on eg. maximum spacing)
- DBSCAN (clustering by density)
- Gaussian Mixtures (ellipsoidal clusters versus k-means' globular clusters)
- etc.

See scikit-learn.org/stable/modules/clustering.html


THE UNIVERSITY OF
SYDNEY

Evaluating Clustering

Cluster Validity

- For supervised classification we have a variety of measures to evaluate how good our model is (accuracy, precision, recall)
- For cluster analysis, the analogous question is how to evaluate the "goodness" of the resulting clusters?
- While clusters are in the eye of the beholder, we may still want to evaluate them…
  - To avoid finding patterns in noise
  - To compare clustering algorithms
  - To compare two sets of clusters
  - To compare two clusters

- http://scikit-learn.org/stable/modules/clustering.html#clustering-evaluation

Measures of Cluster Validity

- Numerical measures that are applied to judge various aspects of cluster validity, are classified into the following three types.
- External Index Measure: Measure the extent to which cluster labels match externally supplied class labels (e.g., v-measure) — how homogeneous the clusters are w.r.t the labels
- Internal Index Measure: Measure the goodness of a clustering structure *without* respect to external information (e.g., SSE) — accuracy of the cluster itself.
- Relative Index Measure: Compare two different clusterings or clusters (often an external or internal index is used)

External Evaluation Measures

- **Homogeneity** ranges from 0 to 1, measuring whether clusters contain data points that are part of a single class
  (analogous to precision)
- **Completeness** ranges from 0 to 1, measuring whether classes contain data points that are part of a single cluster
  (analogous to recall)
- **V-measure** is the harmonic mean of homogeneity and completeness
  (analogous to F1 score)
- http://scikit-learn.org/stable/modules/clustering.html#homogeneity-completeness-and-v-measure

Internal Measure: Sum of Squares Error (SSE)

- For each point, the error is the distance to the nearest cluster
- To get SSE, we square these errors and sum them:

$$SSE = \sum_{i=1}^{K} \sum_{x \in C_i} dist^2(m_i, x)$$

k is the number of clusters, *x* is a data point in cluster C_i and m_i is the representative point (mean) for cluster C_i

$$SSE = \sum_{x=1}^{k} \sum_{x \in C_i} dist^2(m_i, x)$$

Internal Measure: Silhouette Coefficient

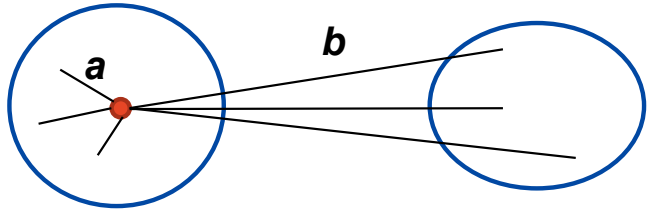
- For an individual point *i*
  - Calculate ***a*** = average distance of *i* to points in its cluster
  - Calculate ***b*** = average distance of *i* to points in the next nearest cluster
  - The silhouette coefficient for a point is then given by

    $s = 1 - a/b$ if $a < b$, (or $s = b/a - 1$ if $a \geq b$, not the usual case)

    $s = 1 - \frac{a}{b}$ if $a < b$

    $b \uparrow \Rightarrow s \uparrow$ | $b \downarrow \Rightarrow s \downarrow$

  - The closer to 1, the better



- Silhouette coefficient for dataset is average across all *i*

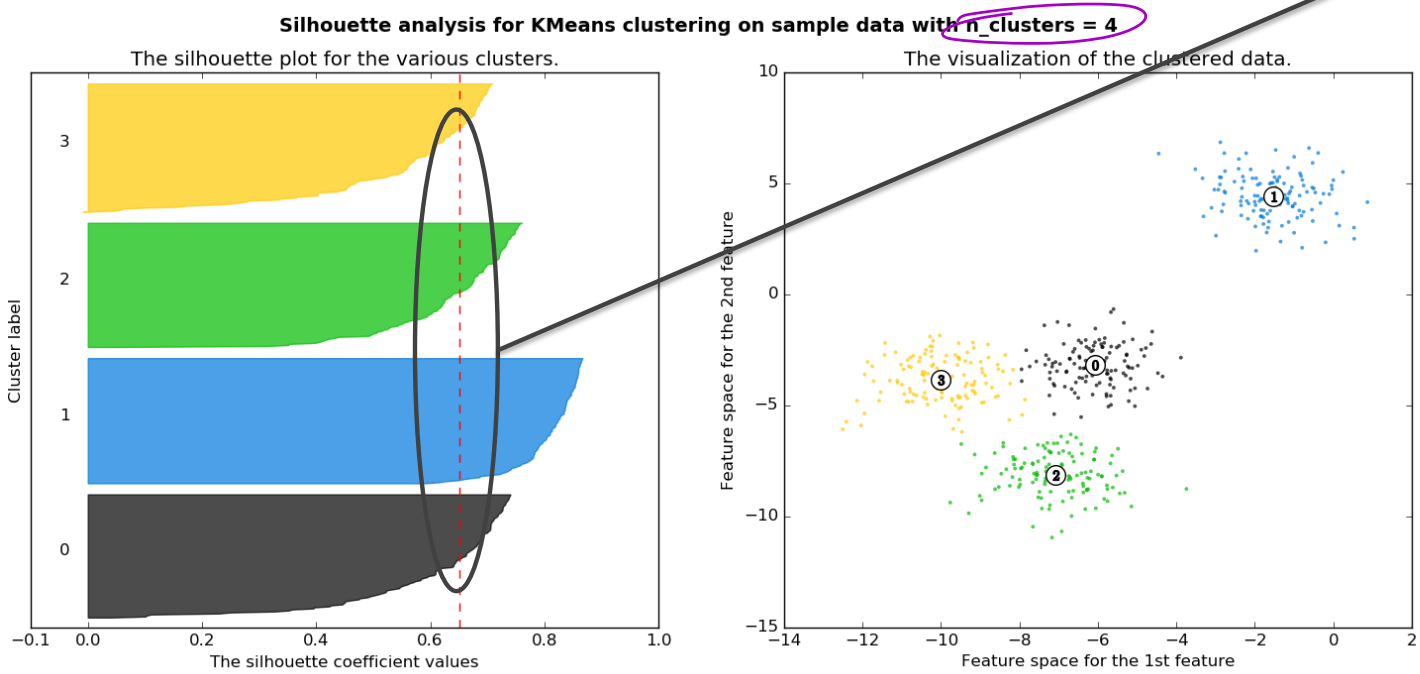
Choosing *k*

- Often we don't know the number of clusters in our data
- Selecting *k* is generally an interactive process
- There are some approaches to aid interactive selection, and sometimes automate it

– Building different clusterings, computing silhouette coefficients then comparing.

Using Silhouettes to choose *k*

High average silhouette indicates points far away from neighbouring clusters

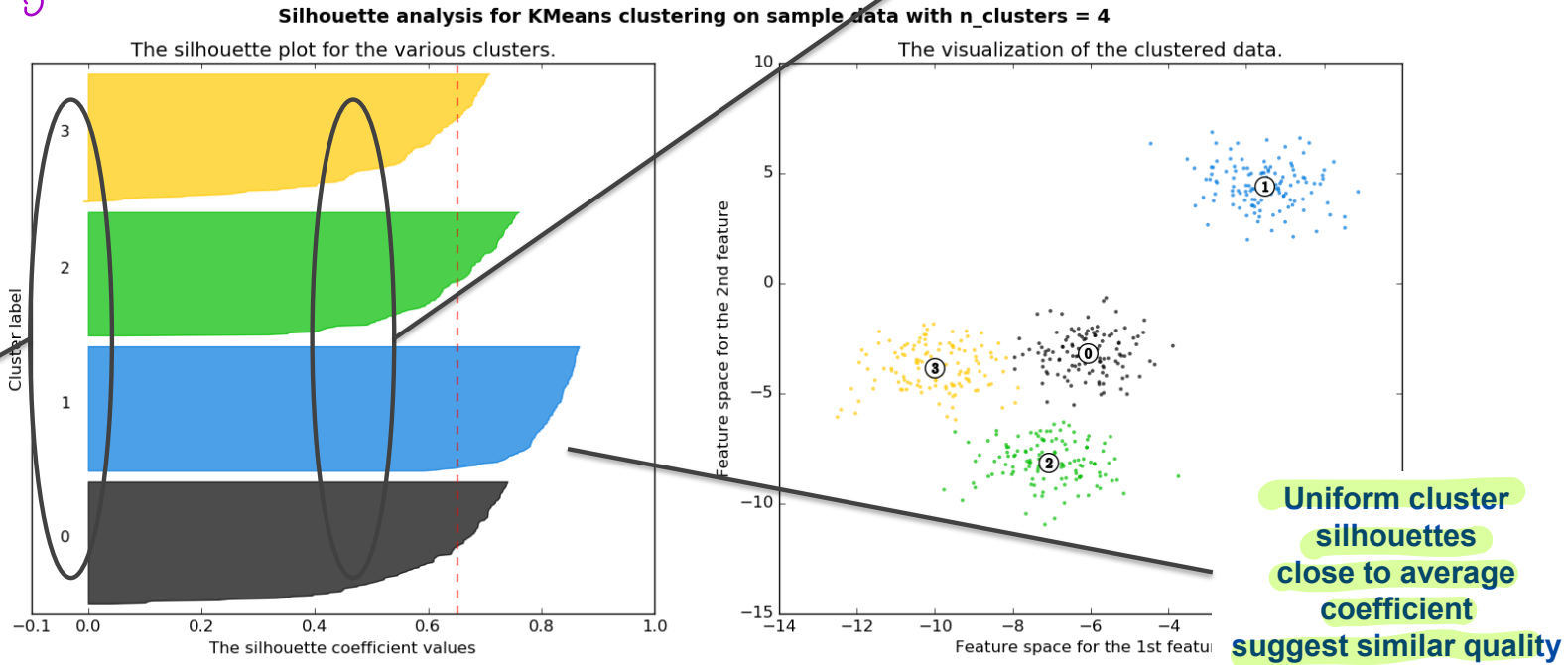


http://scikit-learn.org/stable/auto_examples/cluster/plot_kmeans_silhouette_analysis.html

Using Silhouettes to choose *k*

There's no outlier.

Bar chart showing silhouette values for each item grouped by cluster



Cluster IDs

Uniform cluster silhouettes close to average coefficient suggest similar quality

http://scikit-learn.org/stable/auto_examples/cluster/plot_kmeans_silhouette_analysis.html

Summary – Evaluating Clustering

- Internal and External Evaluation Measures
- Internal Measures
  - Measures goodness of clusterings using just cluster structures
  - SSE and Silhouette Coefficient
- External Measures
  - Measure goodness of clusters using external provided class labels
  - Homogeneity, Completeness, V-measure
- Can be used, e.g., to determine the ideal number of clusters

- http://scikit-learn.org/stable/modules/clustering.html#clustering-evaluation


THE UNIVERSITY OF
SYDNEY

Data Normalisation and Curse of Dimensionality

Pre-Processing for Clustering

- Before we can do effective clustering (or any ML), pre-processing needed

- Data Cleansing
- Data Transformation
- **Data Normalisation**
- **Dimensionality Reduction** / choice or projection of dimensions
  - closely related to choice of **distance metric**
  - we used Euclidean Metric so far (L_2-Norm)
  - but others possible too, eg. Manhatten Distance (L_1-Norm)
  - "Curse of dimensionality"; cf Aggarwal et al.: "On the Surprising Behavior of Distance Metrics in High Dimensional Space", 2001.

Data Normalisation (Feature Scaling)

- normalise the range of independent variables or features of the original data:
  - **Rescaling (min-max normalization)**
    - linear transformation to rescale the range of features to the range [0, 1]

$$x' = \frac{x - \min(x)}{\max(x) - \min(x)}$$

min/max normalisation

  - **Mean normalization**

$$x' = \frac{x - avg(x)}{\max(x) - \min(x)}$$

mean normalisation

  - **Standardisation** (Z-score Normalisation)
    - widely used for normalization in machine learning

$$x' = \frac{x - avg(x)}{\text{stdev}(x)}$$

Z-score normalisation

  - **Scaling to unit length**
    - scale the components of a feature vector by dividing each component by the Euclidean length of the vector

$$x' = \frac{x}{|| x ||}$$

scaling

  - **Log Transformation**

$$x' = \log(x)$$

Log Transformation. normalisation

Distance Measures

- Quantifying similarity between two objects
- Euclidean Space
  - Real-valued dimensions
  - Locations of points in the space
  - “middle” of two points; “dense” region of space
- Non-Euclidean Distance
  - Based on properties of points

Axioms of a Distance Measure

d is a ***distance measure*** if it is a function from pairs of points to real numbers such that:

1. $d(x,y) \geq 0$
2. $d(x,y) = 0$ iff $x = y$
3. $d(x,y) = d(y,x)$
4. $d(x,y) \leq d(x,z) + d(z,y)$ (*triangle inequality*)

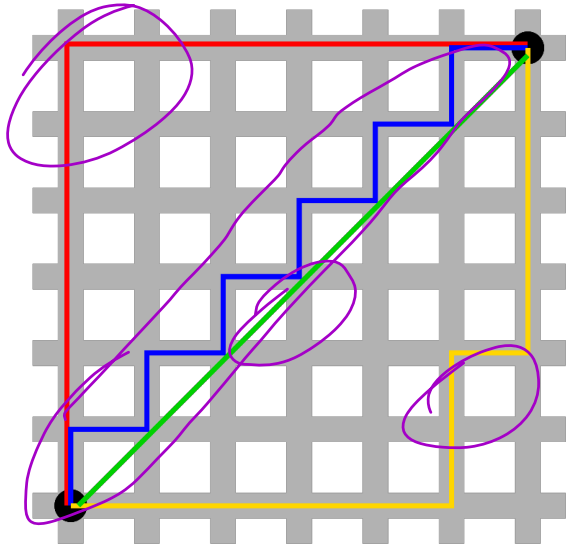
Euclidean vs. Non-Euclidean Distance Measures

- Euclidean
  - L_2 norm (Euclidean distance)
    - Square root of sum of squared differences
  - L_1 norm (Manhattan distance)
    - Sum of absolute differences
  - L_∞ norm (Infinity norm)
    - Maximum of absolute differences

$$d(p,q) = \sqrt{\sum_{i=1}^{n} (q_i - p_i)^2}$$

$$|x_1 - x_2| + |y_1 - y_2|$$

$$|x|_\infty = \max_i |x_i|$$



Manhattan versus Euclidean distance. The red, blue, and yellow lines all have the same length, whereas the green line is the L_2 norm distance.

https://en.wikipedia.org/wiki/Taxicab_geometry

Euclidean vs. Non-Euclidean Distance Measures

- Non-Euclidean
  - Jaccard distance
    - Similarity between two sets
    - 1 – Jaccard similarity
  - Cosine distance
    - Alignment of two vectors
    - 1 – Cosine similarity
  - Edit distance
    - Number of inserts & deletes to change a string into another

also known as Jaccard Index: $J(A,B) = \frac{|A \cap B|}{|A \cup B|}$

also known as cosine similarity: $\cos(\theta) = \frac{A \cdot B}{\|A\| \|B\|} = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2} \sqrt{\sum_{i=1}^{n} B_i^2}}$

The Curse of Dimensionality

- From a theoretical point of view, increasing the number of features should lead to better performance.

- In practice, the inclusion of more features leads to worse performance (i.e., **curse of dimensionality**).
  - "Curse of dimensionality"; cf Aggarwal et al.: "On the Surprising Behavior of Distance Metrics in High Dimensional Space", 2001.

- The more dimensions, the less difference in the distances between data points
  - This complicates clustering or *k*-nearest neighbor classification
- Dimensionality reduction can improve the performance of machine learning

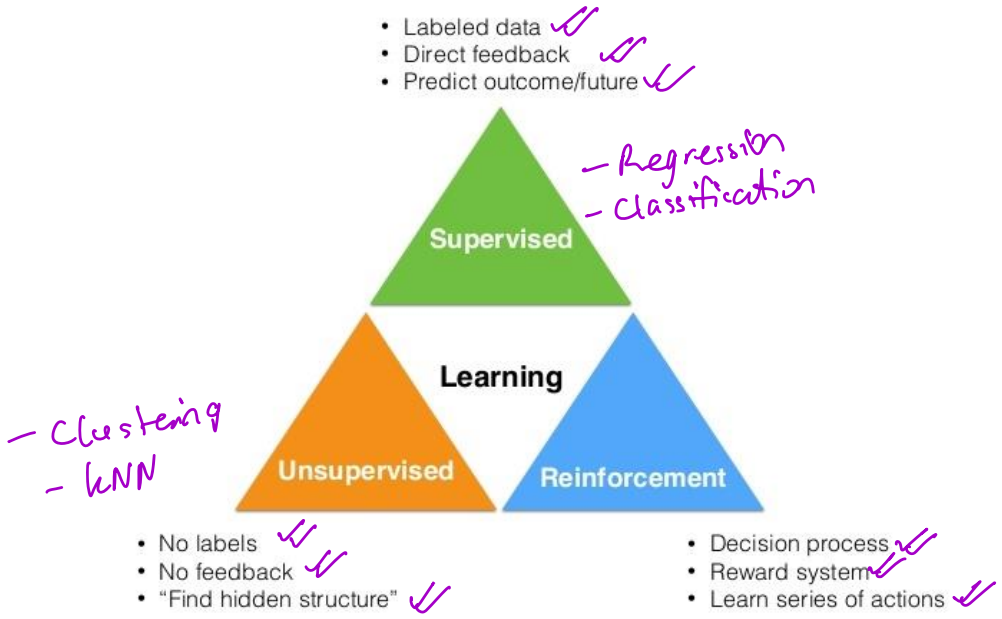
Dimensionality Reduction

- transformation of data from high-dimensional into a low-dimensional space
  - low-dimensional representation retains the meaningful properties of the original data
- Feature Selection
  - Best subset selection (NP problem)
  - Forward/backward stepwise
- Dimension Reduction (Feature Projection)
  - Principal Components Analysis (PCA) → good for clustering
    - Applied only on predictor variables i.e. unsupervised
  - Partial Least Squares
    - chooses predictors related to predicted variables i.e. supervised


THE UNIVERSITY OF
SYDNEY

Supervised Machine Learning

Supervised Machine Learning



http://www.slideshare.net/SebastianRaschka/nextgen-talk-022015/

- Start with data and labels
- Use existing data to develop a model
- Use model to predict outcomes for new data

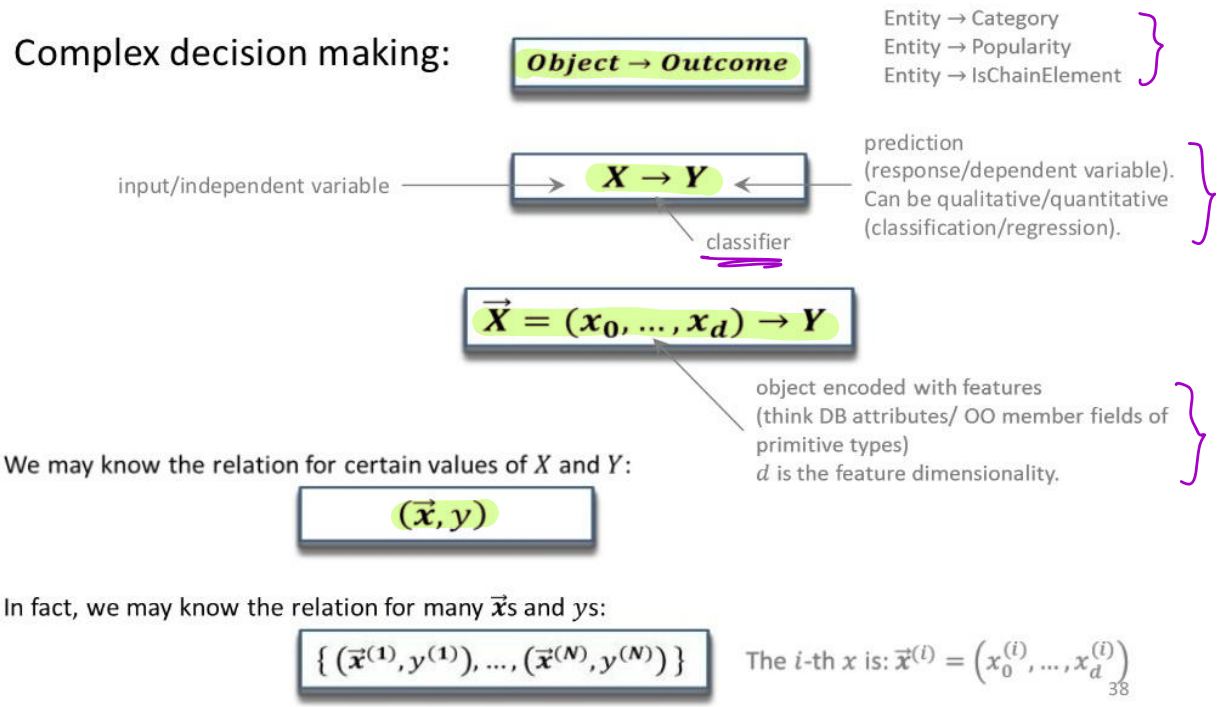
What is a Model?

$$y = f(X)$$

- A **model** is a specification of a mathematical relationship between different variables
- Mapping from features (X) to a numeric value or categorical label (y)

$f: y \rightarrow x$

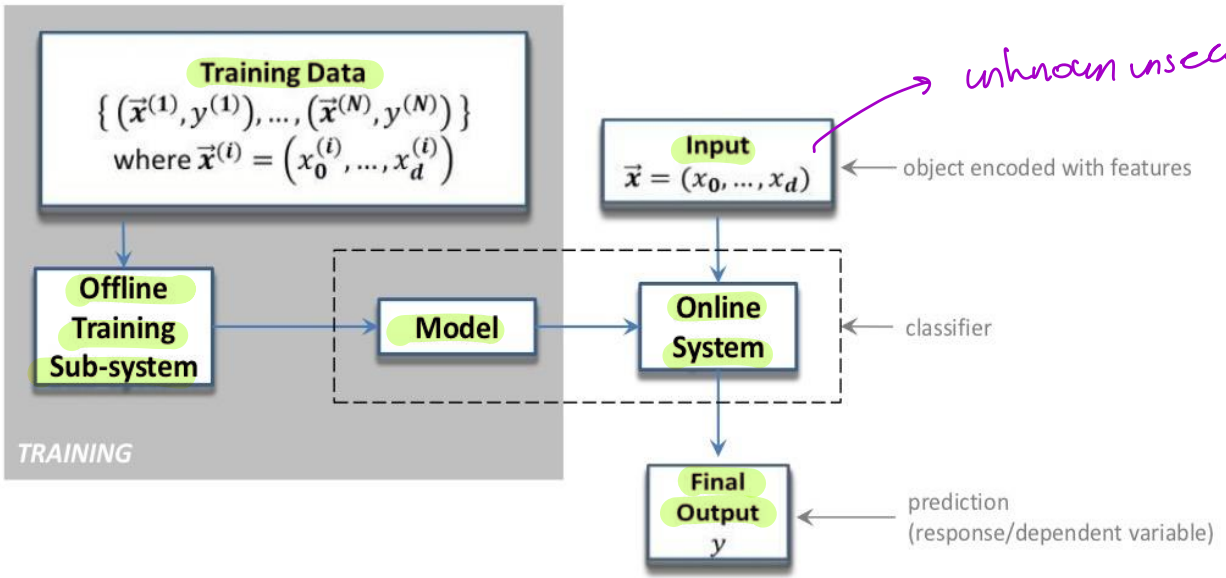
Modelling: Learn a function that maps X→Y



http://www.slideshare.net/Nicolas_Nicolov/machine-learning-14528792

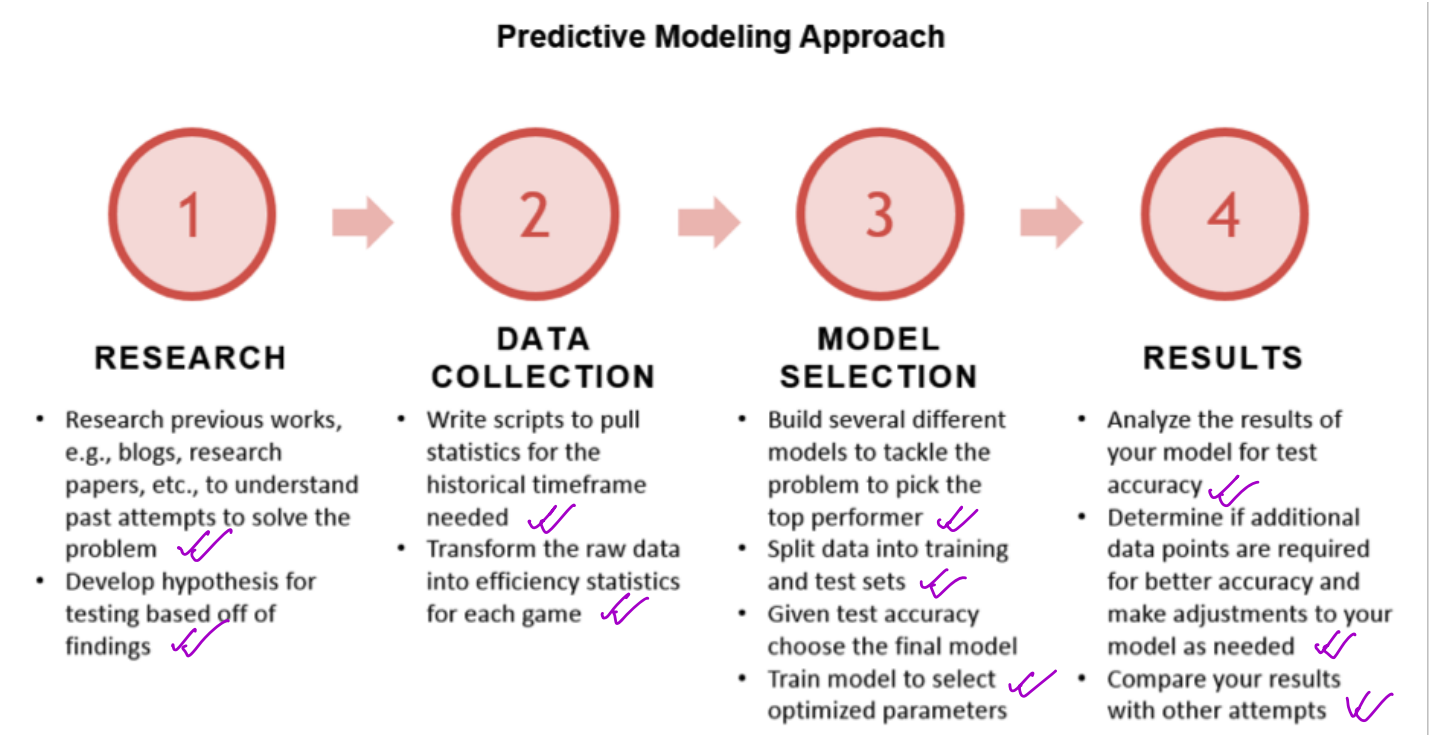
Modelling: Predict Label for new Feature Vectors

$X \rightarrow Y$ $f(X) = Y$



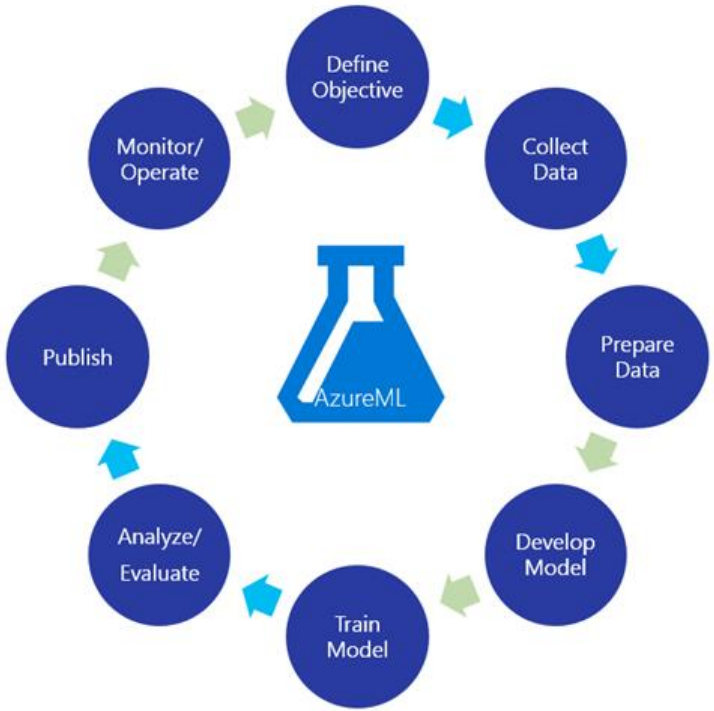
http://www.slideshare.net/Nicolas_Nicolov/machine-learning-14528792

Model Selection and Evaluation



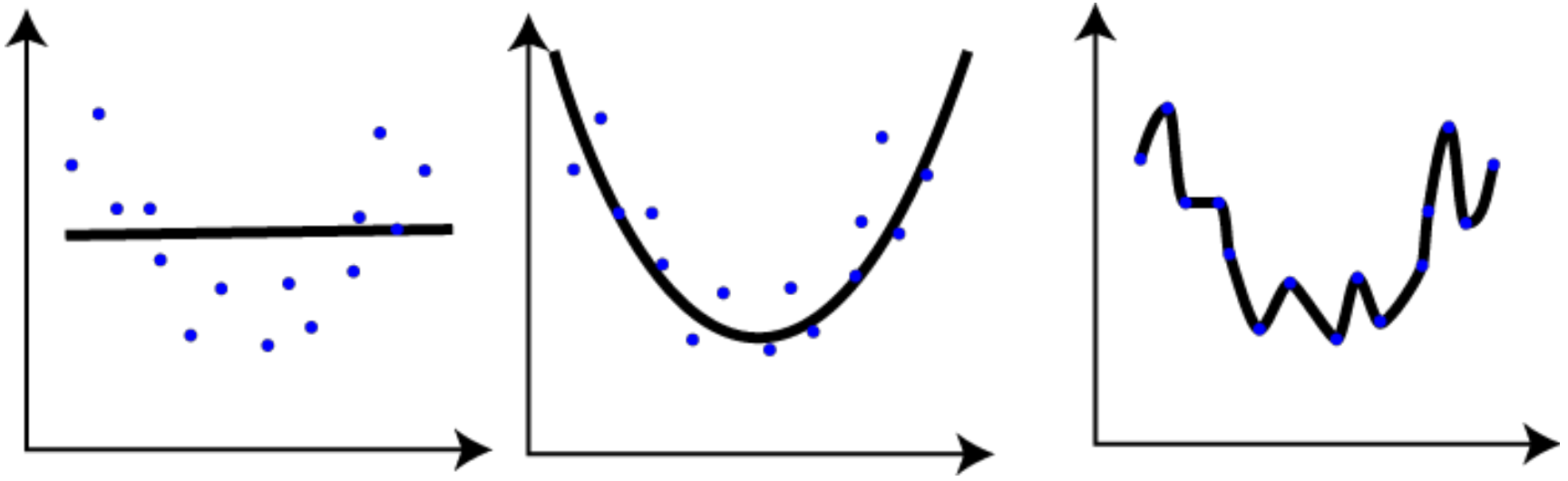
https://www.credera.com/blog/business-intelligence/using-machine-learning-predict-nfl-games/

Machine Learning in Practice



- Model building is one part of a deployment cycle
- Requires serving and monitoring models
  - https://blogs.msdn.microsoft.com/continuous_learning/2014/11/15/end-to-end-predictive-model-in-azureml-using-linear-regression/

Goal is to Build Models that Generalise to Unseen Data

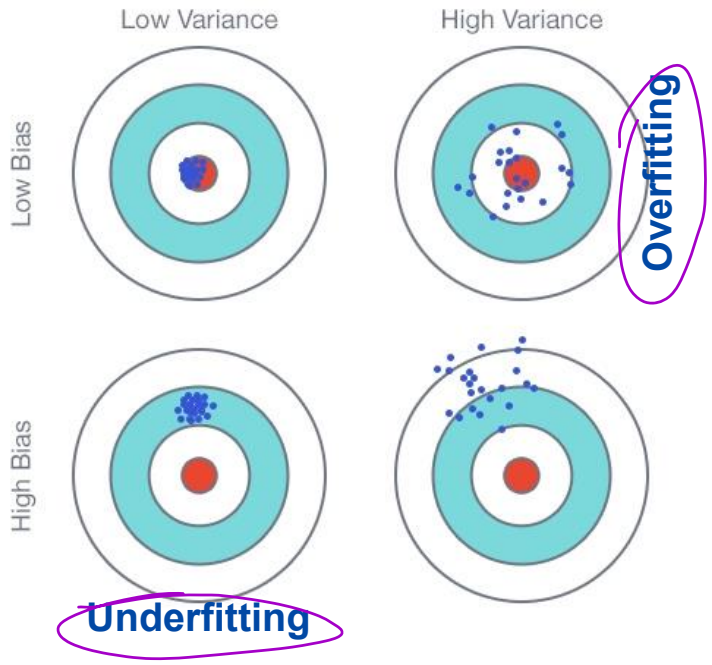


Underfitting

Overfitting

https://shapeofdata.wordpress.com/2013/03/26/general-regression-and-over-fitting/

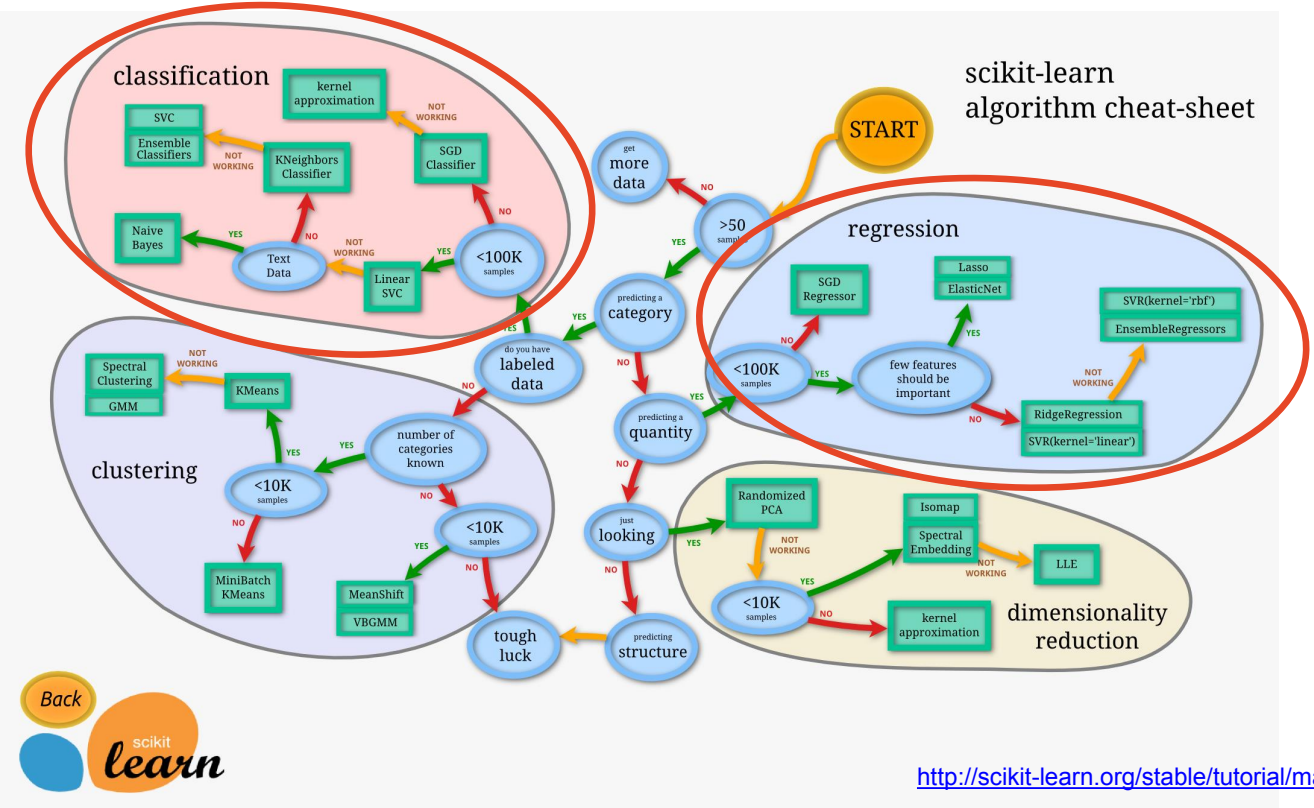
What if we train on many samples from data?



https://medium.com/@mp32445/understanding-bias-variance-tradeoff-ca59a22e2a83

- High **bias** indicates model never fits well so always makes mistakes
- High **variance** indicates different training sets lead to differing results

Machine Learning Map from scikit-learn

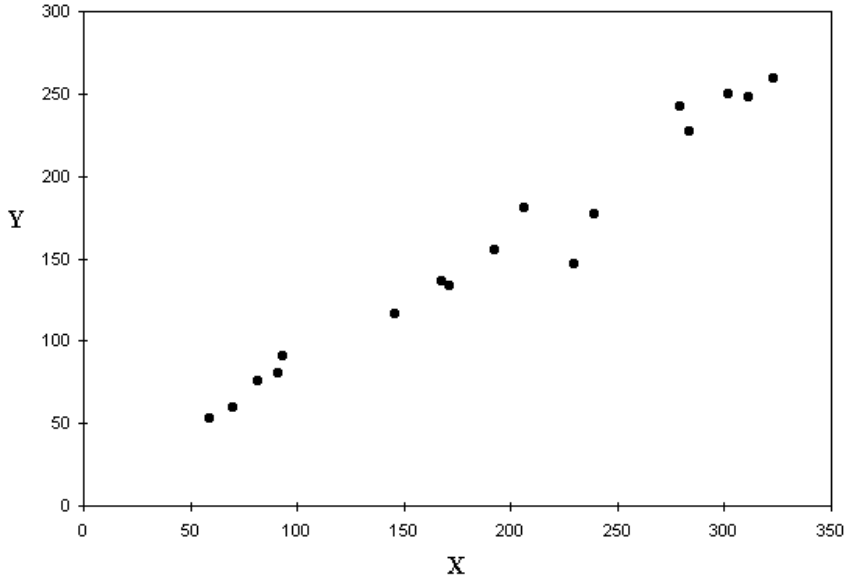


http://scikit-learn.org/stable/tutorial/machine_learning_map/


THE UNIVERSITY OF
SYDNEY

Linear Regression

What is the Relationship between two Variables?



http://home.ku.edu.tr/yihlamur/public_html/Bitirme%20Projesi%20-%20Ger%C3%A7ek%20Data%20D%C3%BCzenlenmi%C5%9F/regression/minitab%20regresion%20hakk%C4%B1nda%20g%C3%BCzel%20bilgiler.htm

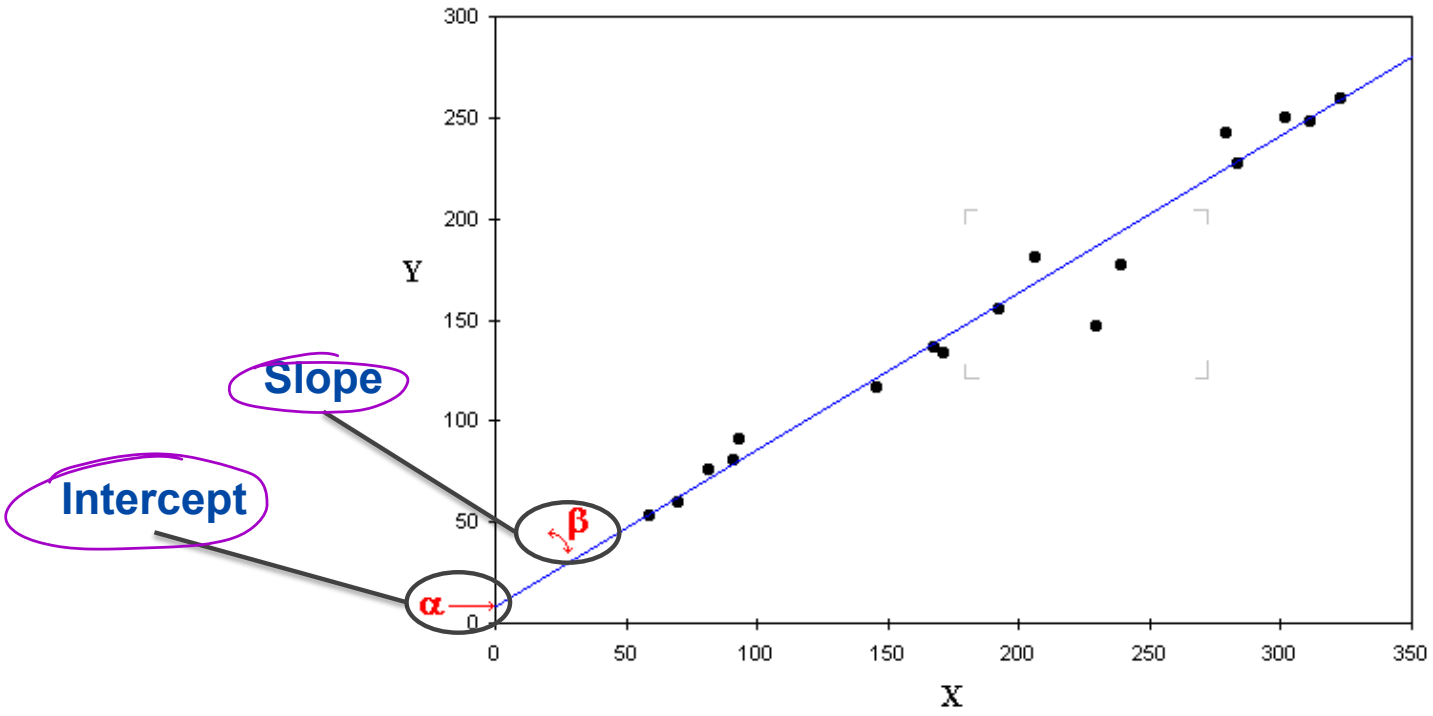
- Correlation measures the strength of the linear relationship
- Often just knowing that there's a relationship isn't enough
- Can we quantify how much one variable depends on the other?

Simple Linear Regression (SLR)

$$Y = \alpha + \beta X + \varepsilon$$

- Method for finding the **line of best fit** between the dependent variable *Y* and the independent variable *X*
- **"Simple"**: only one independent variable

What's the line that explains X⟶Y?



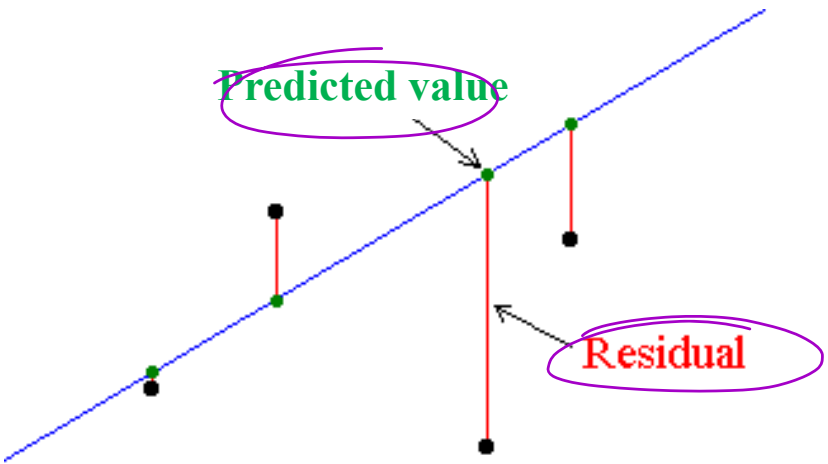
Fitting SLR: learn α and β

$$Y = \alpha + \beta X + \varepsilon$$

- α: Intercept
  (where the line crosses the y axis)
- β: Slope
  (direction and steepness of the line)
- ε: Error
  (error term describing variation of data)

- Y is the dependent variable (response)
- X is the independent variable (predictor)

Fitting SLR: Minimise Sum of Squares Error



http://home.ku.edu.tr/yihlamur/public_html/Bitirme%20Projesi%20-%20Ger%C3%A7ek%20Data%20D%C3%BCzenlenmi%C5%9F/regression/minitab%20regresion%20hakk%C4%B1nda%20g%C3%BCzel%20bilgiler.htm

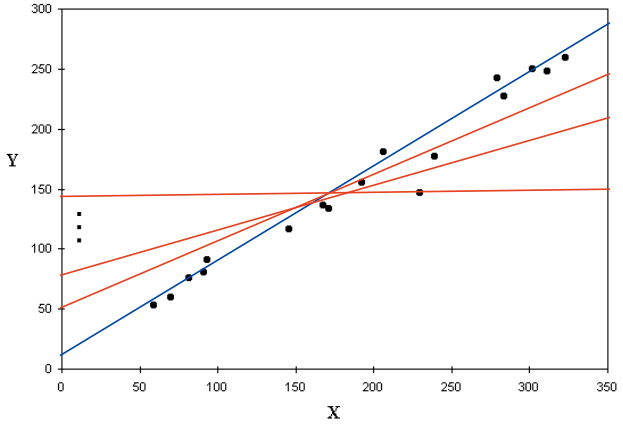
- **Error/residual**: difference between the observed value and predicted value ($y_{actual} - y_{predicted}$)
- Sum of Squares Error:

$$SSE = \sum (y_i - \hat{y}_i)^2$$

Sum Error Square

“Least Squares” Regression

- Intuitively: Find the line that minimizes the squared errors (ie. has the smallest residuals)



- More formally, our goal is:
  Find **α, β** that minimizes $\sum \varepsilon_i^2 = \sum (y_i - \hat{y}_i)^2 = \sum (y_i - (\alpha + \beta x_i))^2$

$$\underset{i}{\operatorname{argmin}} \sum \varepsilon_i^2 = \sum (y_i - \hat{y}_i)^2 = \sum (y_i - (\alpha + (\beta x_i))^2$$

Fitting SLR: Least Squares

proof on STA 5003 notes

- We can resolve the previous least-square equation to:

$$\alpha = \bar{y} - \beta\bar{x} \quad \text{with} \quad \beta = \frac{\sum(x_i - \bar{x})(y_i - \bar{y})}{\sum(x_i - \bar{x})^2} = \frac{\text{corr}(x,y)\ \text{stdev}(y)}{\text{stdev}(x)}$$

Slope of standardized data points with mean 0 and stdev 1

Adjust slope for variation in Y and X

```
def least_squares_fit(x,y):
    """given training values for x and y,
    find the least-squares values of alpha and beta"""
    beta = correlation(x, y) * standard_deviation(y) / standard_deviation(x)
    alpha = mean(y) - beta * mean(x)
    return alpha, beta
```

Intercept is the difference between means of observed and predicted y

Evaluating a Regression Model

- Now that we have determined (α, β) that minimizes SSE on our dataset, the question arises on how well our regression model "fits" the dataset.

- Two metrics are commonly used to measure goodness-of-fit:
  - R-squared (R^2), and
  - the standard error of the regression, often denoted S.

Coefficient of Determination (R^2)

- **R^2**: ratio of **explained variation in y** to **total variation in y**

$$R^2 = 1 - \frac{\sum(y_i - \hat{y}_i)^2}{\sum(y_i - \bar{y})^2} = \frac{SST - SSE}{SST}$$

- Ranges from 0 to 1, with higher values indicating better fit
- Conveys goodness of fit but not precision

(SSE: Sum of Squares Error – sum of the residuals;
SST: Sum of Squares Total – distance of each y_i from the mean)

Standard Error (S)

- Square root of the sum of squared errors divided by N

$$S = \sqrt{\frac{SSE}{N}}$$

- Measure of the prediction accuracy
- Expressed in units of the response variable

Calculating a Prediction Interval from S

- **Prediction interval**: range that should contain the response value of a new observation
- If sample size is large enough then useful rule-of-thumb:

approximately 95% of predictions should fall within

$$\hat{y}_i \pm 2 * S$$

prediction interval, not confidence!

There are functions and libraries available for us to compute this. However, this is just a rule of thumb

Example: Model Acceptance Testing with S

- Suppose we are predicting salary from education level
- Regression model produces R^2=0.761 and *S*=$2k
- Our requirement is that predictions be within $5k

- *S* must be <= $2.5k to produce a sufficiently narrow 95% prediction interval


THE UNIVERSITY OF
SYDNEY

Multiple Linear Regression

and Assessing Fit and Standard Error

Multiple Linear Regression

$$Y = \alpha + \beta_1 X_1 + \beta_2 X_2 + \ldots + \beta_k X_k + \varepsilon$$

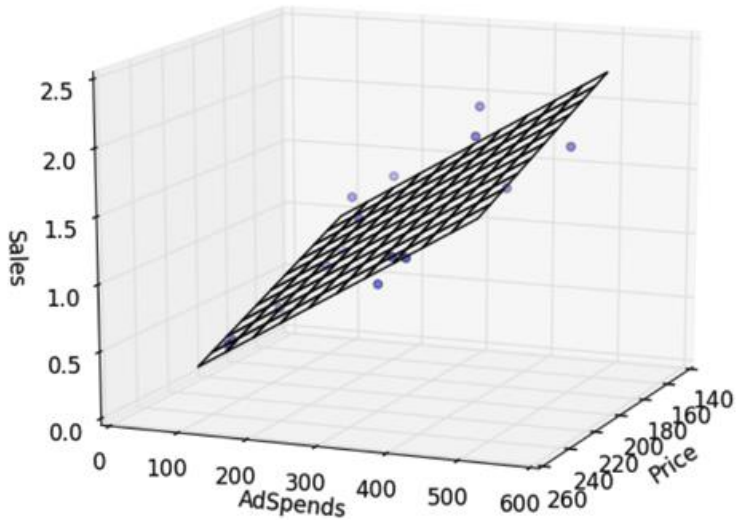
- Explain the relationship between:
  - **two or more** explanatory variables (X_1 to X_k)
  - one response variable Y

Example: Multiple Linear Regression

- With two variables, means fitting a response plane into a 3-dimensional space

p predictors
p+1 dimensional space.

- While we can at most visualize 3D, linear regression can be applied to arbitrary vector spaces
  - $X = (x_1, \ldots, x_k)$ vector of explanatory vars
  - Y response variable



https://medium.com/@dhwajraj/python-regression-analysis-part-4-multiple-linear-regression-ed09a0c31c74

Machine Learning in scikit-learn

we predict on the test dataset, but fit Lin Reg on training data.

```
from sklearn.linear_model import LinearRegression
lm = LinearRegression()
_ = lm.fit(X_train, Y_train)
Y_test = lm.predict(X_test)
```

There are other ways to fit Lin Reg in Python programming

- **Estimator**: a Python object that implements the methods fit(X, y) and predict(T)
  - eg. **sklearn.linear_model.LinearRegression**
- **Fit(X, y)**: fits a model to the training data X, y
  - **X**: feature vectors
  - **y**: labels (here: numerical value)
- **Predict(T)**: predict labels for new data T

Assessing Fit and Standard Error

We can use the same methods than with Simple Linear Regression to assess goodness-of-fit:

R-squared (R^2), and
the standard error S of the regression.

```
# We use the score method to get r-squared
print('\nR-squared:', lm.score(X_train, Y_train))

# We can also calculate the standard error
stderr = math.sqrt(np.mean((Y_train - lm.predict(X_train))**2))
print('\nStandard error:', stderr)
```

Example: Predicting House Prices

```
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
import math
import numpy as np

# load Scikit.learn's example dataset of Boston house prices
from sklearn.datasets import load_boston
boston = load_boston()

# First let's create a train and test split
X_train, X_test, Y_train, Y_test = train_test_split(boston.data, boston.target, test_size=0.33,
                                                    random_state=5) # so we get the same results

# Now let's fit a model
lm = LinearRegression()
_ = lm.fit(X_train, Y_train)

print('Intercept:', lm.intercept_)
print('Coefficients:\n', lm.coef_)
print('\nR-squared:', lm.score(X_train, Y_train))
```

Typical stuff...

X y

lm.score(X, y) → R^2 value.

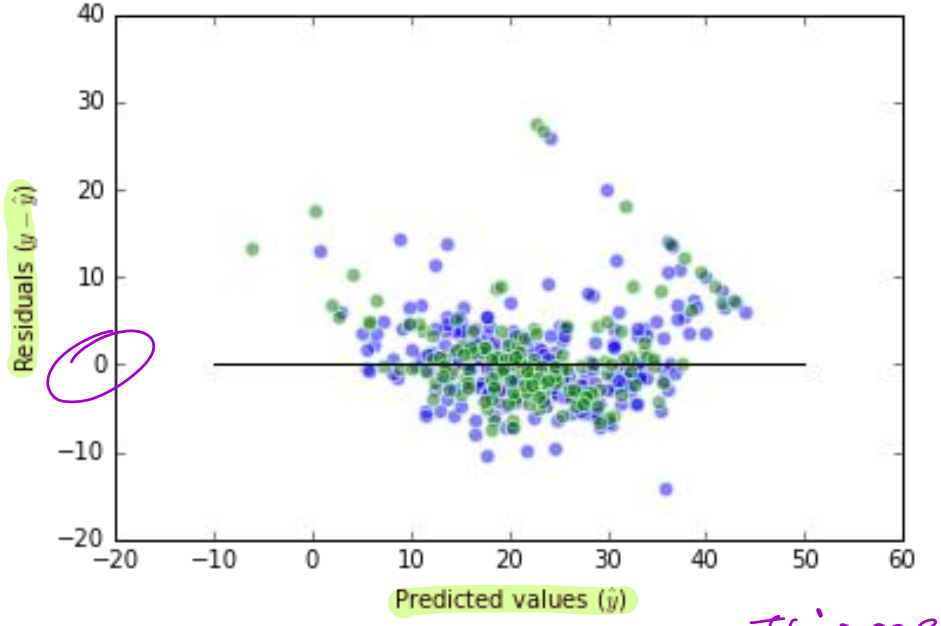
coefficients for predictors.

```
Intercept: 32.858932634085924
Coefficients:
 [-1.56381297e-01  3.85490972e-02 -2.50629921e-02  7.86439684e-01
 -1.29469121e+01  4.00268857e+00 -1.16023395e-02 -1.36828811e+00
  3.41756915e-01 -1.35148823e-02 -9.88866034e-01  1.20588215e-02
 -4.72644280e-01]

R-squared: 0.7551332741779998
```

```
:Attribute Information (in order):
    - CRIM     per capita crime rate by town
    - ZN       proportion of residential land zoned for lots over 25,000 sq.ft.
    - INDUS    proportion of non-retail business acres per town
    - CHAS     Charles River dummy variable (= 1 if tract bounds river; 0 otherwise)
    - NOX      nitric oxides concentration (parts per 10 million)
    - RM       average number of rooms per dwelling
    ...
```

Visually Assessing Good Fit: Residual Plots



This one isn't too bad...

Residual Plots indicate good fit if…

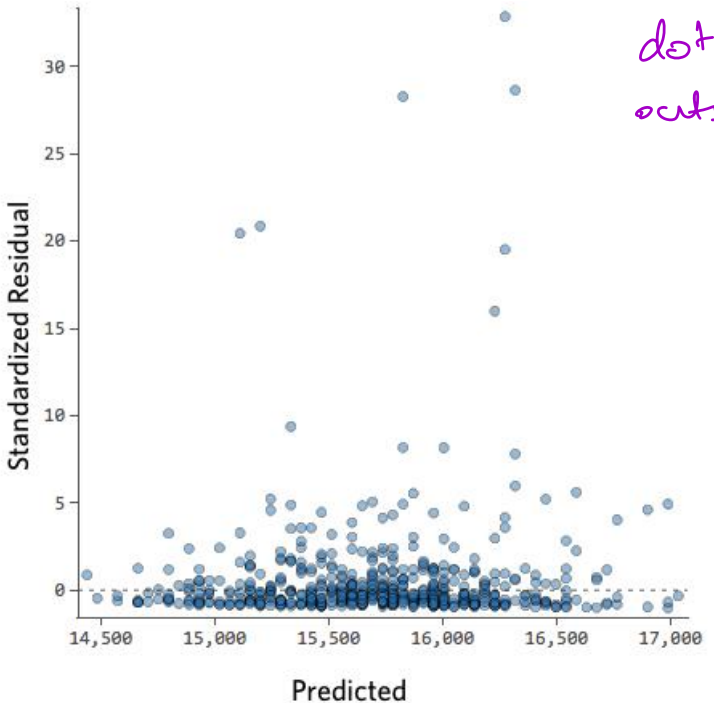
- symmetrically distributed, clustering towards middle of plot
- there aren't any clear patterns
- they cluster around y=0
- green training points, blue predicted points

y means residual at 0.

Bad Residuals

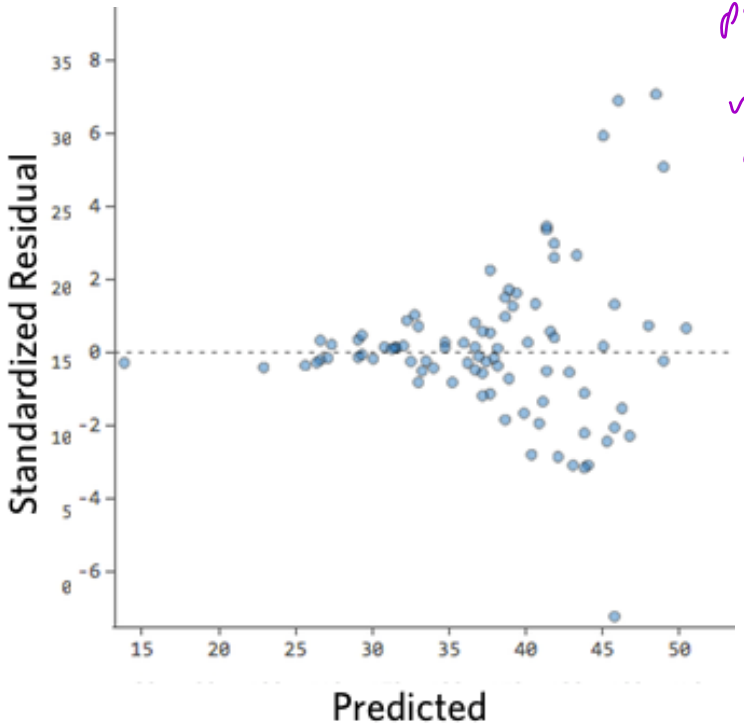
y-axis off balance

because of dotted line outside of 0.



Inconsistent variance

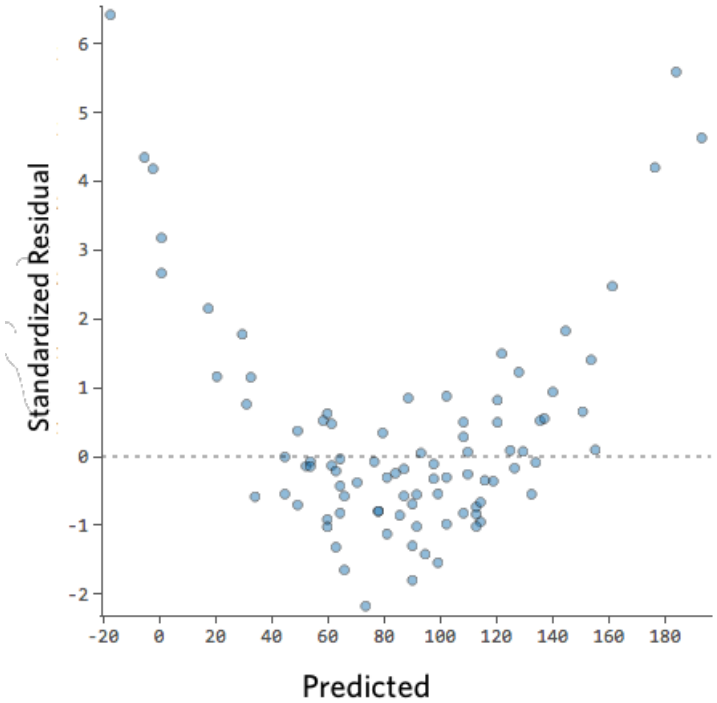
data points are not well clustered



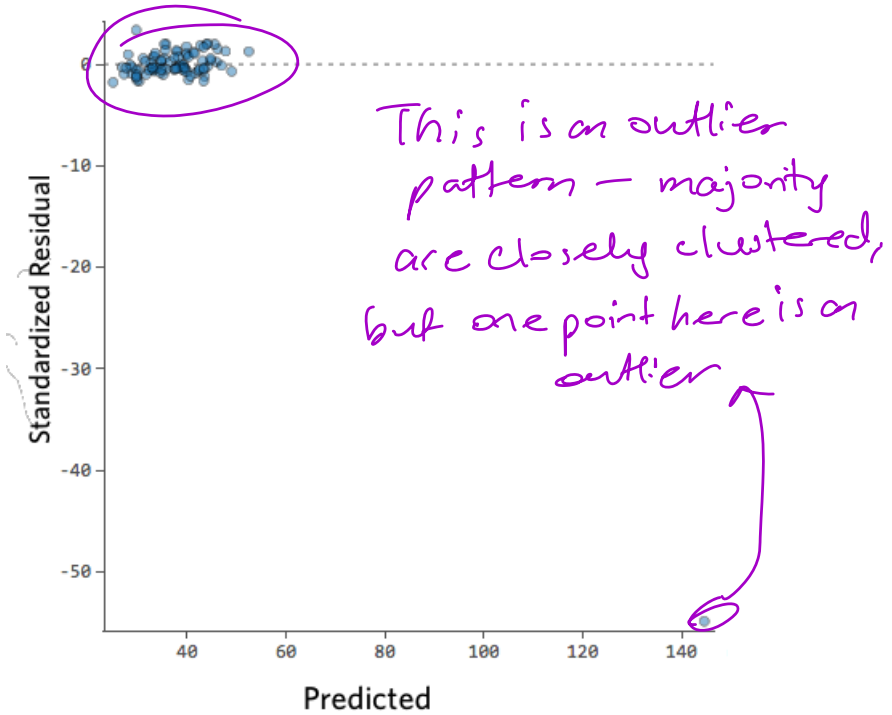
http://docs.statwing.com/interpreting-residual-plots-to-improve-your-regression/

More Bad Residuals

Non linear



Outlier



http://docs.statwing.com/interpreting-residual-plots-to-improve-your-regression/


THE UNIVERSITY OF
SYDNEY

Logistic Regression

Classification vs Regression

recently covered in STAT5003

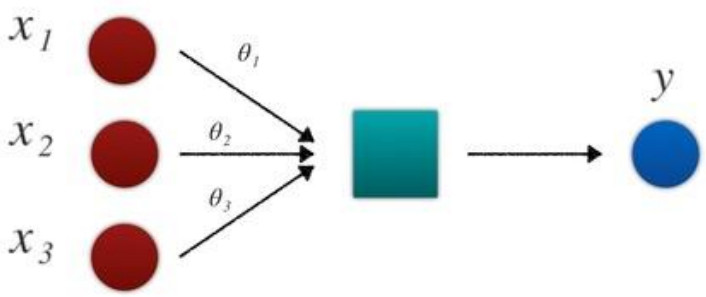
- Classification assigns a class to each example
- Output is a discrete / categorical variable
- E.g., predict whether tumor is harmful or not harmful

- Regression assigns a numerical value
- Output is a continuous variable (real value)
- E.g., predict house price

Logistic Regression

predictors

labels



- Predict the probability of some categorical label
- E.g., given
  - amount of debt
  - late payment count

  predict the probability of defaulting on a loan

http://www.toshistats.net/101-4-logistic-regression/

- Binary classification
- Multi-class classification

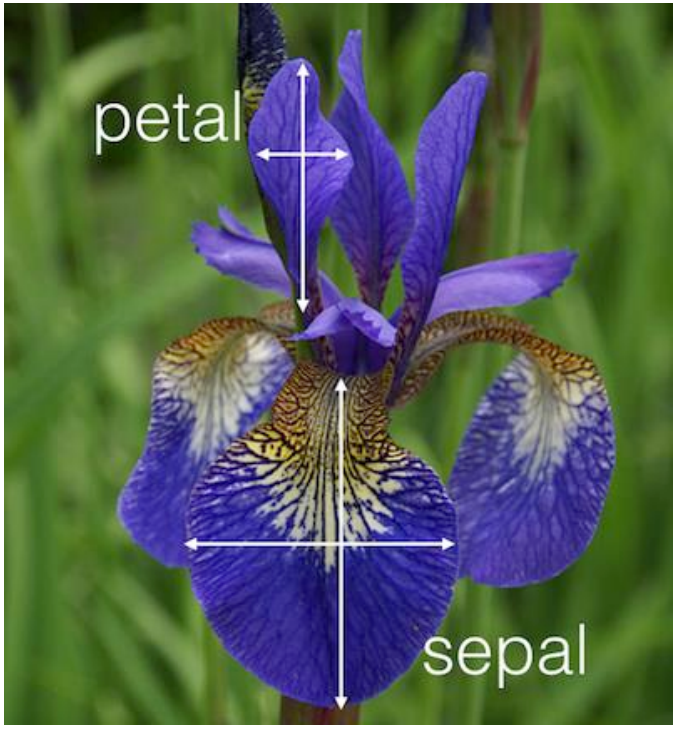
Multinomial Logistic Regression

- Generalises logistic regression to multi-class problems
- E.g., predicting iris type:

- given measurements of individual flowers

Labels ....

Iris Measurement: Petal vs Sepal



- Features:
  - Petal height
  - Petal width
  - Sepal height
  - Sepal width

  } Numerical features
- Target
  - three classes of types of Iris plants

3 labels (or classes)

- https://en.wikipedia.org/wiki/Iris_flower_data_set

Split data and train a classifier in scikit-learn

```
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression

# First let's create a train and test split
X_train, X_test, Y_train, Y_test = train_test_split(iris.data, iris.target, test_size=0.33)


# Now let's fit a model
logreg = LogisticRegression()
_ = logreg.fit(X_train, Y_train)
```

```
Intercept: [  8.77541301   1.60989219 -10.38530519]
Coefficients:
 [[-0.40882379  0.8471058  -2.20548191 -0.96488141]
 [ 0.60527175 -0.44107978 -0.16035026 -0.8906139 ]
 [-0.19644796 -0.40602602  2.36583217  1.85549531]]

Predicted type of first five organisms from test split: [1 2 2 0 2]
Actual type of first five organisms from test split: [1 2 2 0 2]
```

matches exactly the same.

(0=setosa, 1=versicolor, 2=virginica)

Although we don't use R^2 as the eval metric for classification → explore other metrics later

Evaluating Classification

```
from sklearn.metrics import classification_report
import pandas as pd
key=', '.join(['{}={}'.format(i,name) for i,name in enumerate(iris.target_names)])
print('Classification report ({}):\n'.format(key))
print(classification_report(Y_test, logreg.predict(X_test)))
```

This is not necessary to remember.

```
Classification report (0=setosa, 1=versicolor, 2=virginica):

              precision    recall  f1-score   support

           0       1.00      1.00      1.00        16
           1       1.00      0.94      0.97        17
           2       0.94      1.00      0.97        17

    accuracy                           0.98        50
   macro avg       0.98      0.98      0.98        50
weighted avg       0.98      0.98      0.98        50
```

precision and recall are in percentage values

Prevent Overfitting with Regularisation

- Large feature spaces introduce problems with **overfitting**
- **Regularisation** adds a penalty that gets larger as coefficients (β) get larger
  - Can be incorporated into the cost function
  - The more weight we give to the error term, the more we discourage large coefficients

- Can also address overfitting by eliminating features (either manually or via model selection)

*Regularisation is a tuning technique for overfitting, besides eliminating features.

Regularisation Parameters in scikit-learn

- Penalty
  - **l1 (lasso)**: estimates sparse coefficients; equivalent to feature selection
  - **l2 (ridge)**: minimises coefficients; equivalent to setting a prior
- C
  - Inverse of regularisation strength
  - Small values specify stronger regularisation

Selecting Model Parameters with Grid Search

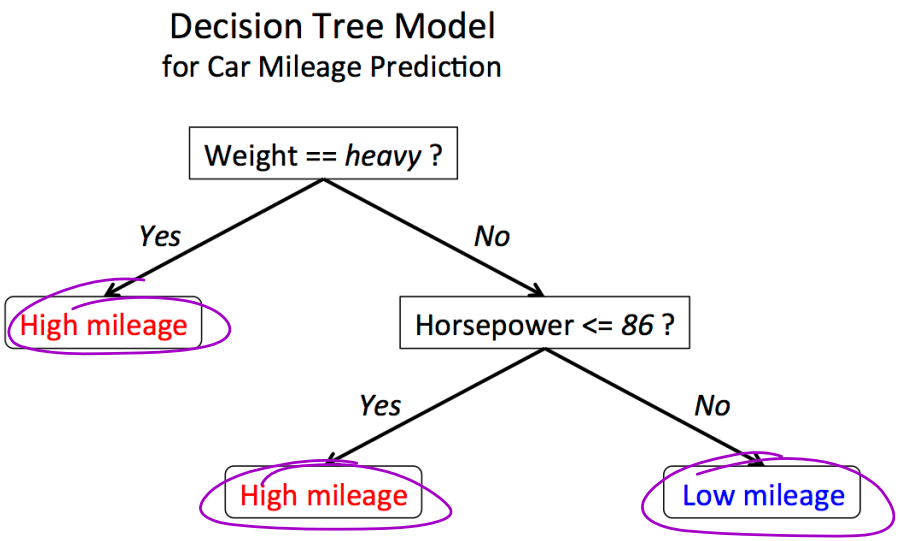
- Parameters like penalty and regularisation strength are not learnt from data by default
- Can be set using exhaustive search through combinations of specified possible values
- Perform n-fold cross validation for each combination
- In scikit-learn:
  - from sklearn.grid_search import GridSearchCV
  - http://scikit-learn.org/stable/auto_examples/model_selection/grid_search_digits.html


THE UNIVERSITY OF
SYDNEY

Decision Trees

Supervised Machine Learning for Classification using Decision Trees

Decision Tree Classification (and Regression)



- Maps observations to a target value
- Can be viewed as hierarchy of if/else statements
- Resulting model is intuitive and interpretable (+can be visualized)
- Ensembles of simple trees can do very well

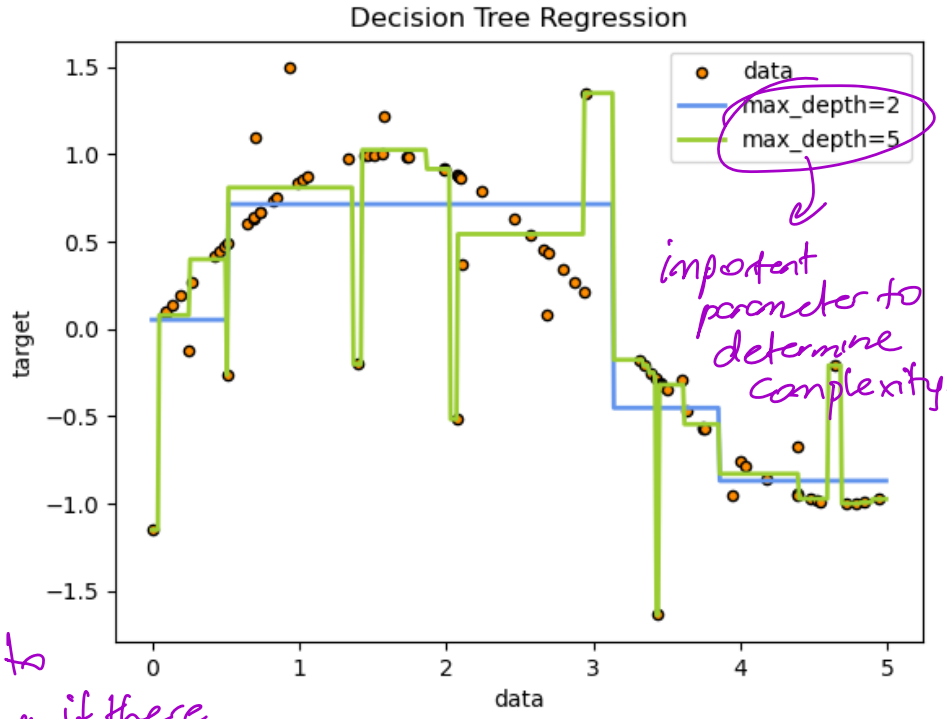
https://databricks.com/blog/2014/09/29/scalable-decision-trees-in-mllib.html

Decision Trees (cont'd)

not helpful for representing continuous functions.

Piece-wise functions of decision trees.

- Basically learn a piece-wise function that approximates data
- Disadvantages
  - Prone to overfitting by creating over-complex models if depth not limited
  - Can be unstable due to small data variations -> ensembles of trees
  - Predictions are not continuous
  - If some classes dominate in data, can produce biased trees



important parameter to determine complexity.

Prone to overfitting, if there are a lot of depth in a tree.

https://scikit-learn.org/stable/auto_examples/tree/plot_tree_regression.html

Training a Decision Tree Classifier in scikit-learn

- DecisionTreeClassifier
- Can be parameterized, e.g.:
  - max_depth
    - the maximum depth of the tree
  - criterion
    - entropy: choose splits that minimise total uncertainty
    - gini: choose splits that minimise misclassification
  - splitter
    - best: choose the optimal threshold for each feature
    - random: choose the best random threshold for each feature

As usual, split data into training and test sets first. Then:

```
from sklearn.tree import DecisionTreeClassifier

# Let's fit a model
tree = DecisionTreeClassifier(max_depth=2)
_ = tree.fit(X_train, Y_train)
```

Evaluating a Decision Tree Classifier

- To assess the performance of a decision tree model, you can follow the general guidelines for Model Evaluation
  - Determine the number of true and false positives, and true and false negatives
  - Compute from those the Accuracy, Precision and Recall metrics
    - Recall is also called the ***Sensitivity*** of a Decision Tree
  - Also helpful in the context of decision trees:
    - **Specificity** (TNR - True Negative Rate): *TNR = TN / (TN + FP)*
    - **Miss Rate** (FNR – False Negative Rate): *FNR = FN / (FN + TP)*
    - **False Discovery Rate** (FDR): *FDR = FP / (FP + TP)*

$$TNR = \frac{TN}{TN+FP}$$

$$FNR = \frac{FN}{FN+TP}$$

$$FDR = \frac{FP}{FP+TP}$$

Comparing Classifiers on a single Test Split

- How to compare two decision tree classifiers?
- Use McNemar's test to compare two classifiers over a single test split
  - Split dataset into training and test data sets
  - Train both classifiers on the training dataset
  - Using the test data, determine the number of instances misclassified by each classifier

- H_0: Two classifiers have the same error rate
- H_A: One is better (assume whichever has higher accuracy)

- http://sci2s.ugr.es/keel/pdf/algorithm/articulo/dietterich1998.pdf

McNemar's Test

- Given:
  - n1: number of instances where Classifier1 is correct, but Classifier2 is not
  - n2: number of instances where Classifier2 is correct, but Classifier1 is not
- Then:

$$\frac{(|n1 - n2| - 1)^2}{(n1 + n2)}$$

chi-squared

follows a χ^2 distribution with 1 degree of freedom if the null hypothesis is correct. If in your evaluation it doesn't, you can reject H_0

Tips and Tricks

- Compare ML models to the simplest baseline first; Iterate
- Best strategy is often a simpler model with more/better data
- If working with high-dimensional data, beware of overfitting
  - Feature selection and/or use suitable hyperparameters to make model less sensitive
- Always test on held-out data that hasn't been used for training


THE UNIVERSITY OF
SYDNEY