MACHINE LEARNING FUNDAMENTALS:A CASE STUDY APPROACH

Summer Internship Report Submitted in partial fulfillment

of the requirement for under graduate degree of

Bachelor of Technology

In

COMPUTER SCIENCE ENGINEERING

By

TANMAY AGARWAL

221710305058

Under the Guidance of

Mr.SDV Prasad

Assistant Professor

Department Of Computer Science&Engineering

GITAM School of Technology

GITAM (Deemed to be University)

Hyderabad-502329

July 2020

DECLARATION

I submit this industrial training work entitled **"**MACHINE LEARNING FUNDAMENTALS:A CASE STUDY APPROACH" to GITAM (Deemed to be University), Hyderabad in partial fulfillment of the requirements for the award of the degree of "**Bachelor of Technology**" in "**Computer Science&Engineering** ". I declare that it was carried out independently by me under the guidance of **Mr.SDV Prasad**, Asst. Professor, GITAM (Deemed to be University), Hyderabad, India.

The results embodied in this report have not been submitted to any other University or Institute for the award of any degree or diploma.

Place: HYDERABAD TANMAY AGARWAL

Date: 221710305058

GITAM (Deemed to be University)

Hyderabad-502329, India

CERTIFICATE

This is to certify that the Industrial Training Report entitled **““**MACHINE LEARNING FUNDAMENTALS:A CASE STUDY APPROACH”**”** is being submitted by TANMAY AGARWAL in partial fulfillment of the requirement for the award of **Bachelor of Technology in Computer Science&Engineering** at GITAM (Deemed to be University), Hyderabad during the academic year 2020-21.

It is faithful record work carried out by him at the **Computer Science&Engineering** , GITAM school of technology Hyderabad Campus under my guidance and supervision.

Mr.SDV Prasad
Assistant Professor
Department of CSE

Dr.S.PHANI KUMAR
Professor and HOD
Department of CSE

UNIVERSITY of WASHINGTON

COURSE CERTIFICATE

08/13/2020

tanmay agarwal

has successfully completed

Machine Learning Foundations: A Case Study Approach

a MOOC from the University of Washington and offered through Coursera

Emily Fox
Amazon Professor of Machine Learning
Statistics

Carlos Guestrin
Amazon Professor of Machine Learning
Computer Science and Engineering

Verify at coursera.org/verify/5BXUFK4QDFFE

Coursera has confirmed the identity of this individual and their participation in the course.

Machine Learning : A case study approach

Tanmay Agarwal

221710305058

Contents

1 Introduction

Humans can expand their knowledge to adapt the changing environment. To do that they must "learn". Learning can be simply defined as the acquisition of knowledge or skills through study, experience, or being taught. Although learning is an easy task for most of the people, to acquire new knowledge or skills from data is too hard and complicated for machines. Moreover, the intelligence level of a machine is directly relevant to its learning capability.

The study of machine learning tries to deal with this complicated task. In other words, machine learning is the branch of artificial intelligence that tries to find an answer to this question: how to make computer learn?

When we say that the machine learns, we mean that the machine is able to make predictions from examples of desired behavior or past observations and information. More formal definition of machine learning by Tom Mitchell is A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P, if its performance at tasks in T, as measured by P, improves with experience E. The definition also indicates the main goal of machine learning: the design of such programs

1.1 Taxonomy Of Machine Learning

The Machine Learning process starts with inputting training data into the selected algorithm. Training data being known or unknown data to develop the final Machine Learning algorithm. The type of training data input does impact the algorithm, and that concept will be covered further momentarily.

To test whether this algorithm works correctly, new input data is fed into the Machine Learning algorithm. The prediction and results are then checked.

If the prediction is not as expected, the algorithm is re-trained multiple numbers of times until the desired output is found. This enables the Machine Learning algorithm to continually learn on its own and produce the most optimal answer that will gradually increase in accuracy over time.

There are three main types of machine learning algorithms :

1.Supervised Learning

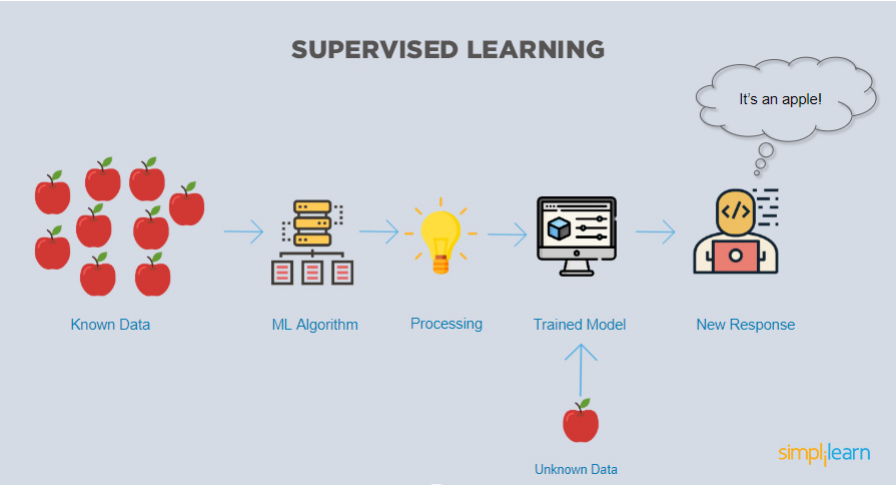
2.Unsupervised Learning

3.Reinforced Learning

1.1.1 Supervised Learning

In supervised learning, we use known or labeled data for the training data. Since the data is known, the learning is, therefore, supervised, i.e., directed into successful execution. The

input data goes through the Machine Learning algorithm and is used to train the model. Once the model is trained based on the known data, you can use unknown data into the model and get a new response.

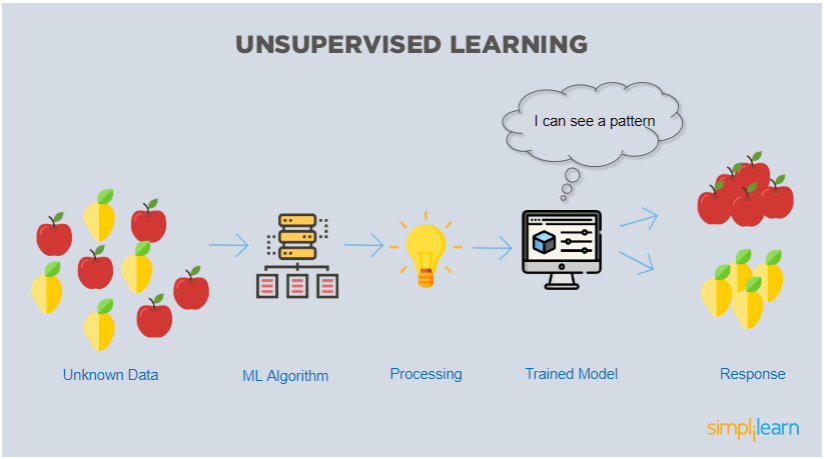


In the case of the below figure, the model tries to figure out whether the data is an apple or another fruit. Once the model has been trained well, it will identify that the data is an apple and give the desired response.

1.1.2 Unsupervised Learning

In unsupervised learning, the training data is unknown and unlabeled meaning that no one has looked at the data before. Without the aspect of known data, the input cannot be guided to the algorithm, which is where the unsupervised term originates from. This data is fed to the Machine Learning algorithm and is used to train the model.

The trained model tries to search for a pattern and give the desired response. In this case, it is often like the algorithm is trying to break code like the Enigma machine but without the human mind directly involved but rather a machine.



In this case, the unknown data consists of apples and pears which look similar to each other. The trained model tries to put them all together so that you get the same things in similar groups.

1.1.3 Reinforced Learning

Like traditional types of data analysis, here, the algorithm discovers data through a process of trial and error and then decides what action results in higher rewards. Three major components make up reinforcement learning: the agent, the environment, and the actions. The agent is the learner or decision-maker, the environment includes everything that the agent interacts with, and the actions are what the agent does.

Reinforcement learning occurs when the agent chooses actions that maximize the expected reward over a given time. This is easiest to achieve when the agent is working within a sound policy framework.

1.2 Prerequisites For Machine Learning

For those interested in learning beyond what is Machine Learning, a few requirements should be met to be successful in pursual of this field. These requirements include:

1. Basic knowledge of programming and scripting languages
2. Intermediate knowledge of statistics and probability
3. Basic knowledge of linear algebra
4. Knowledge of how to clean and structure raw data

2 Objectives

Machine learning in business and other areas such as healthcare and governmental departments is not simply another term for AI (Artificial Intelligence). While AI is the umbrella term given for machines emulating human abilities, machine learning is a specific branch of AI where machines are trained to learn how to process and make use of data; another description often used is ‘machine intelligence’.

The objective of machine learning in business is not only for effective data collection, but to make use of the ever increasing amounts being gathered by manipulating and analysing it without heavy human input.

Machine intelligence enables complex and larger data to be processed and analysed along with the desired results being achieved such as determining customer trends, detecting

fraud, spotting buying trends and other primary objectives.

Machine learning in business therefore offers an important commercial benefit in being able to make the best use of your data.

Indeed, a key objective of machine learning is to enable you to keep up with those competitors already making best use of their data to maximise business opportunities.

Most commercial and non-commercial organisations benefit from machine learning, so it's highly likely that some form of machine intelligence can be put to use in your business.

2.1 Why use a case study approach?

we're gonna learn about this data to intelligence pipeline is by examining a number of case studies that are gonna ground the methods that we present in real world applications. And that's one of the really unique features of this course.

In our first case study, we're gonna look at predicting house values. So, the intelligence we're deriving is a value associated with some house that's not on the market. So, we don't know what it's value is and we wanna learn that from data.

We're gonna look at other houses and look at their house sales prices to inform the house value of this house we're interested in. And in addition to the sales prices, we're gonna look at other features of the houses. Like how many bedrooms the houses have. Bathrooms, number of square feet, and so on.And what we're gonna do, our machine learning method is something that's gonna relate the house attributes to the sales price.

Because if we can learn this model, this relationship from our house level features to the observed sales price, then we can use that for predicting on this new house. We take its house attribute and predict its house sales price. And this method is called regression.

In our second case study, we're gonna explore a sentiment analysis task where we have reviews of some restaurants. So for example in this case, it says the sushi was awesome, the food was awesome, but the service was awful. And we wanna take this review and be able to classify whether it had positive sentiment. It was a good review, thumbs up or a negative sentiment, thumbs down. Well, we're gonna look at a lot of other reviews.

So, we're gonna look at the text of the review and the rating of the review. In order to understand what's the relationship here, for classification of this sentiment. So, for example in this case, maybe we might analyze the text of this review in terms of how many times it uses the word awesome versus how many times

it uses the word awful. And from these other reviews that we have, we're gonna learn some decision boundary between based on the balance of usage of these words whether it's a positive or negative review. And the way we learn that from these other reviews is based on the ratings associated with that text. And so this method is called a classification method.

In our third case study, we're gonna do a document retrieval task where here, what we wanna do, the intelligence we're deriving is an article or a book or something like this

that's of interest to our reader. And the data that we have is a huge collection of possible articles that we could recommend. And what we're gonna do, in this case, is we're gonna try and find structure in this data based on groups of related articles.

Such as, maybe there's a collection of articles about sports and world news and entertainment and science. And if we find this structure and annotate our corpus, our collection of documents with these types of labels which we don't have ahead of time, we're trying to infer this from the data.

Then we can use this for very rapid document retrieval because if I'm sitting here currently reading some article about world news, then maybe, if I wanna retrieve another article, I already know which articles to search over.And this type of approach is called clustering.

In our fourth case study, we're gonna do this really interesting thing that's called collaborative filtering that's had a lot of impact in many domains in the last decade. Specifically, we're gonna look at doing product recommendation, where you take your past purchases and trying to use those to recommend some set of other products you might be interested in purchasing.

So in this case, the data that we're gonna use to derive this intelligence for product recommendation is we'd like to understand what's the relationship between what you bought before and what you're likely to buy in the future. And to do this, we're gonna use other users' purchase histories. And possibly, features of those users.

But the key idea here is we're gonna take this data and we're gonna arrange it into this customers by products matrix where the squares here indicate products that a customer actually purchased. So those are products that are liked by that customer. And from this matrix, we're gonna learn features about users and features about products.

And once we learn those features about users and products from this data that I've described. We can think about using those features to see how much agreement there is between what the user likes, different attributes the user likes and whether the product is actually about those attributes. So in the example I'm showing here, maybe a user is a mom, has certain features that are similar to other users that are also moms. And from that, we can infer things about products. What are attributes about.

For example, baby products that are of interest to moms. And we're using that information to form our recommendations. And this type of approach going from this matrix, this customers products matrix into these learned features about users and products is called matrix factorization.

2.2 Final Goal

Our final goal after learning this in to build,test and train different forms of machine leaning models. We would like tho implement Machine leaning in an easy and effictive way. This course will help us to solve real life problems using machine learning. Finally, halfway between supervised and unsupervised learning lies semi-supervised learning. In this case, the algorithm is provided with both unlabeled as well as labeled data. Techniques of this

category are particularly useful when available data are incomplete and to learn representations.

As supervised learning is by far the most widespread form of machine learning in materials science, we will concentrate on it in the following discussion. Workflow applied in supervised learning. One generally chooses a subset of the relevant population for which values of the target property are known or creates the data if necessary.

This process is accompanied by the selection of a machine learning algorithm that will be used to fit the desired target quantity. Most of the work consists in generating, finding, and cleaning the data to ensure that it is consistent, accurate, etc. Second, it is necessary to decide how to map the properties of the system, i.e., the input for the model, in a way that is suitable for the chosen algorithm.

This implies to translate the raw information into certain features that will be used as inputs for the algorithm. Usually this entails the adjustment of hyperparameters that control the training process, structure, and properties of the model. The data are split into various sets. Ideally, a validation dataset separate from the test and training sets is used for the optimization of the hyperparameters.

3 Outcomes

The outcome of this 6 weeks long course us to make my fundamentals in machine learning much stronger. By the end of this course I was able to build my own machine learning models, train and test data, manipulate datasets, solve real life problems based on machine learning, build and deploy artificial neural netwroks in the python environment.

Machine learning is ubiquitous in the industry these days. Organizations around the world are scrambling to integrate machine learning into their functions and new opportunities for aspiring data scientists are growing multifold. But we have noticed a huge gap between what the industry needs and what's on offer right now. Quite a large number of people are not clear about what machine learning is.

By end of this course, I was not only understand what is machine learning but also it's different types, its ever-growing list of applications, the latest machine learning developments, the top experts in machine learning, among various other things.

4 Technologies Learnt

This Course was an extensive 6 week course which helped me learn and understand machine learning with a unique case study approach. I will be dividing each week's content as sub-sections within this section.

4.1 Environment Setup And Sframes Introduction

In the first week we were introduced to the basics of machine learning. We were taught why we should be learning it from a case study point of view. In the later days of the first week we were instructed to download the softwares and were given an introduction on how to use Sframes. The software used in week 1 includes :

1.Python 3.7

2.Jupyter notebook

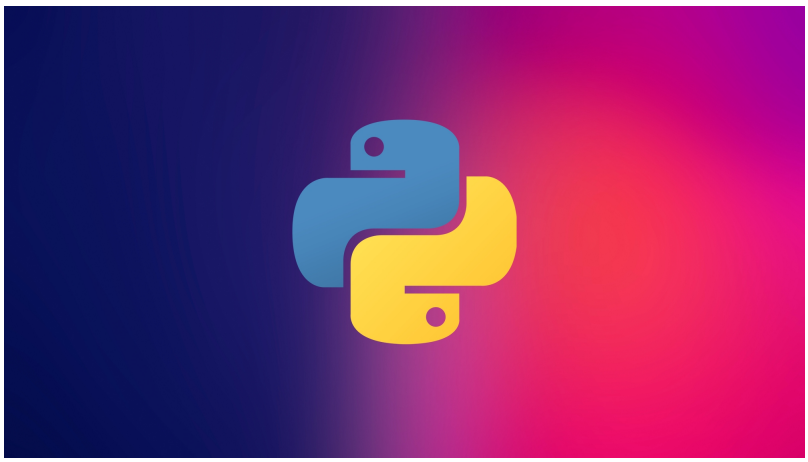
3.Sframes package

4.Turi Create

4.1.1 Python Language

Python is an interpreted, object-oriented, high-level programming language with dynamic semantics. Its high-level built in data structures, combined with dynamic typing and dynamic binding, make it very attractive for Rapid Application Development, as well as for use as a scripting or glue language to connect existing components together.

Python's simple, easy to learn syntax emphasizes readability and therefore reduces the cost of program maintenance. Python supports modules and packages, which encourages program modularity and code reuse. The Python interpreter and the extensive standard library are available in source or binary form without charge for all major platforms, and can be freely distributed.



Often, programmers fall in love with Python because of the increased productivity it provides. Since there is no compilation step, the edit-test-debug cycle is incredibly fast.

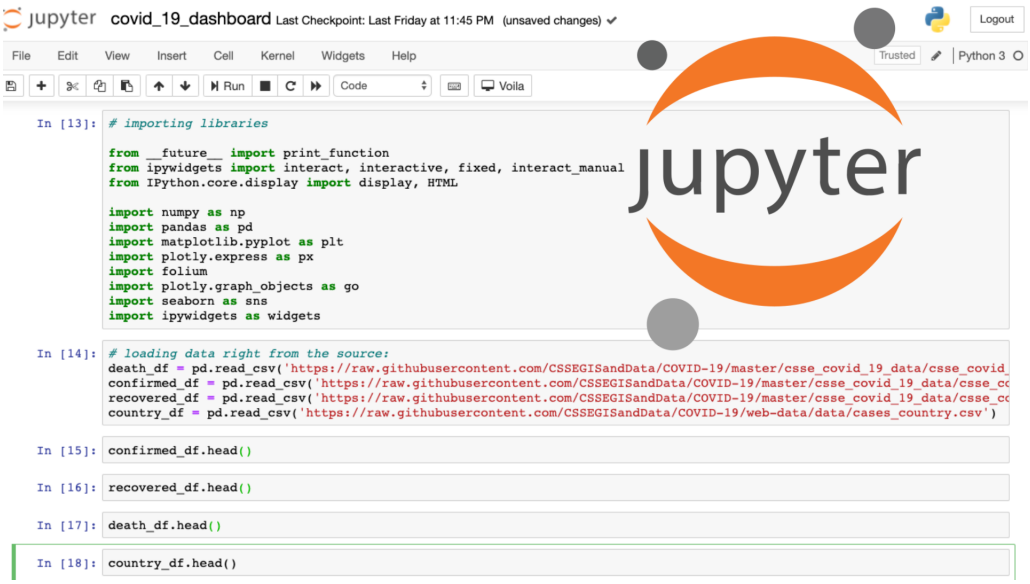
Debugging Python programs is easy: a bug or bad input will never cause a segmentation fault. Instead, when the interpreter discovers an error, it raises an exception. When the program doesn't catch the exception, the interpreter prints a stack trace. A source level debugger allows inspection of local and global variables, evaluation of arbitrary expressions, setting breakpoints, stepping through the code a line at a time, and so on.

The debugger is written in Python itself, testifying to Python's introspective power. On the other hand, often the quickest way to debug a program is to add a few print statements to the source: the fast edit-test-debug cycle makes this simple approach very effective.

4.1.2 Jupyter Notebok

The Jupyter Notebook is an open-source web application that allows you to create and share documents that contain live code, equations, visualizations and narrative text.

Uses include: data cleaning and transformation, numerical simulation, statistical modeling, data visualization, machine learning, and much more.



4.1.3 Sframes

SFrame is an scalable, out-of-core dataframe, which allows you to work with datasets that are larger than the amount of RAM on your system.Both an SFrame and a DataFrame are

Python data structures for representing data sets. In both, a row represents a record and a column represents a variable. In both, records and variables can be reached using indexes.

An SFrame was defined inside Graphlab Create. It can scale to big data (hence the ”S” in the name). An SFrame is column-immutable, but entire columns can be added and subtracted from it. It supports fast out-of-core operations. It has an open source (BSD license) implementation.

PassengerId	Survived	Pclass	Sex	Age	Fare
1	0	3	male	22.0	7.25
2	1	1	female	38.0	71.2833
3	1	3	female	26.0	7.925
4	1	1	female	35.0	53.1
5	0	3	male	35.0	8.05
6	0	3	male	None	8.4583

A DataFrame is defined inside the pandas library. It is designed to work inside RAM. It is mutable, and supports a wider range of operations than an SFrame.

4.1.4 Turi Create

Turi Create is an open source toolset for creating Core ML models, for tasks such as image classification, object detection, style transfers, recommendations, and more. Learn how you can use Turi Create to build models for your apps.

Turi Create simplifies the development of custom machine learning models. You don’t have to be a machine learning expert to add recommendations, object detection, image classification, image similarity or activity classification to your app.

Easy-to-use: Focus on tasks instead of algorithms Visual: Built-in, streaming visualizations to explore your data Flexible: Supports text, images, audio, video and sensor data Fast and Scalable: Work with large datasets on a single machine Ready To Deploy: Export models to Core ML for use in iOS, macOS, watchOS, and tvOS apps

4.2 Regression And It's Implementation

In week 2 we started with learning the most basic Machine Learning model which is a linear regressor. It is a supervised machine learning model.

1.Linear Regression Modelling

2.Evaluating Regression models

4.2.1 Linear Regression Modelling

In statistics, linear regression is a linear approach to modeling the relationship between a scalar response (or dependent variable) and one or more explanatory variables.The case of one explanatory variable is called simple linear regression. For more than one explanatory variable, the process is called multiple linear regression.his term is distinct from multivariate linear regression, where multiple correlated dependent variables are predicted, rather than a single scalar variable.

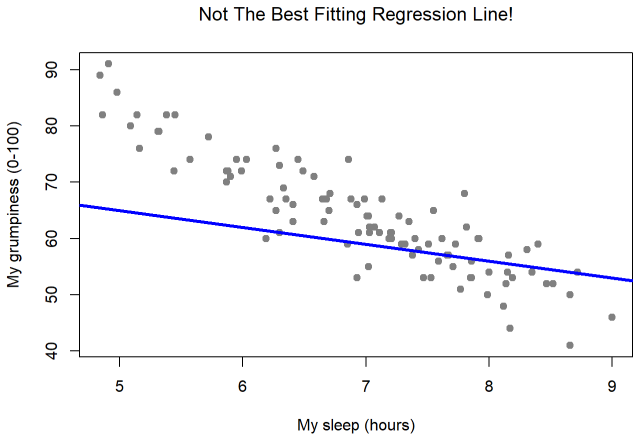
In linear regression, the relationships are modeled using linear predictor functions whose unknown model parameters are estimated from the data. Such models are called linear models.

Most commonly, the conditional mean of the response given the values of the explanatory variables (or predictors) is assumed to be an affine function of those values; less commonly, the conditional median or some other quantile is used. Like all forms of regression analysis, linear regression focuses on the conditional probability distribution of the response given the values of the predictors, rather than on the joint probability distribution of all of these variables, which is the domain of multivariate analysis.

Linear regression has many practical uses. Most applications fall into one of the following two broad categories:

1.If the goal is predicting, forecasting or error reduction

2.If the goal is to explain variation in response



Given a data set

$$[y_i, x_{i1}, x_{i_2}...., x_{i3}]$$

of n statistical units, a linear regression model assumes that the relationship between the dependent variable y and the p-vector of regressors x is linear. This relationship is modeled through a disturbance term or error variable ϵ an unobserved random variable that adds

noise to the linear relationship between the dependent variable and regressors. Thus the model takes the form

$$Y_i = \beta_0 + \beta_1 x_i + ... + \beta_p x_p = x_i^T \beta + \epsilon$$

The above formula is just an expanded form of the simple lne formula

$$y = mx + c$$

our intercept and our slope are the parameters of our model. And so, to be very explicit here, we're gonna write this function, this linear function here, with the subscript W, that indicates that this function is specified by parameters. W being the set of W_0 and W_1. Okay, so this is the line that we fit through the data. But a question is, which line is the right line or a good line to use for a given data set.

4.2.2 Evaluating Regression Modelling

Validation and Evaluation of a Data Science Model provides more colour to our hypothesis and helps evaluate different models that would provide better results against our data. These are the metrics that help us evaluate our models.There are three main errors (metrics) used to evaluate models :

1.Mean Absolute Error

2.Mean Squared Error

3.R2 score

Lets take an example where we have some points. We have a line that fits those points. When we do a summation of the absolute value distance from the points to the line, we get Mean absolute error. The problem with this metric is that it is not differentiable. Let us translate this into how we can use Scikit Learn to calculate this metric.

$$\text{MAE} = \frac{1}{n} \sum_{t=1}^{n} |e_t|$$

Mean Squared Error solves differentiability problem of the MAE. Consider the same diagram above. We have a line that fits those points. When we do a summation of the square of distances from the points to the line, we get Mean squared error.

$$\text{MSE} = \frac{1}{n} \sum_{t=1}^{n} e_t^2$$

Let us take a naive approach by taking an average of all the points by thinking of a horizontal line through them. Then we can calculate the MSE for this simple model. R score answers the question that if this simple model has a larger error than the linear regression

model. However, it terms of metrics the answer we need is how much larger. The R score answers this question. R2 score is $1-$(Error from Linear Regression Model/Simple average model).

$$\text{RMSE} = \sqrt{\frac{1}{n}\sum_{t=1}^{n} e_t^2}$$

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.

4.2.3 Summary of Linear Regression

Regression models describe the relationship between variables by fitting a line to the observed data. Linear regression models use a straight line, while logistic and nonlinear regression models use a curved line. Regression allows you to estimate how a dependent variable changes as the independent variable(s) change.

Simple linear regression is used to estimate the relationship between two quantitative variables.

Simple linear regression is a parametric test, meaning that it makes certain assumptions about the data. These assumptions are:

1.Homogeneity of variance (homoscedasticity): the size of the error in our prediction doesn't change significantly across the values of the independent variable.

2.Independence of observations: the observations in the dataset were collected using statistically valid sampling methods, and there are no hidden relationships among observations.

3.Normality: The data follows a normal distribution.

4.The relationship between the independent and dependent variable is linear: the line of best fit through the data points is a straight line (rather than a curve or some sort of grouping factor). If your data do not meet the assumptions of homoscedasticity or normality, you may be able to use a nonparametric test instead, such as the Spearman rank test.

4.3 Classification Based Models

In this week we took a deeper dive into Classification models. We studied about the different types of classification models and the various methods used to test them. We also looked the ways to evalute these different classification models

1.Classifiction modelling

2.Linear Classifiers

3.Evaluating Classification models

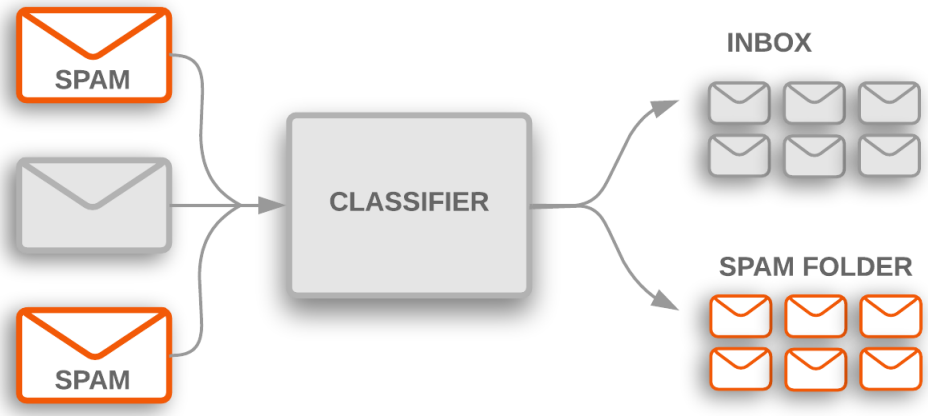
4.3.1 Classification Modelling

Classification is one of the most common areas of machine learning, one of the most used, and useful areas of machine learning. And perhaps, one of those most intuitive ones as well.

So things like figuring out whether your email's spam or not, or whether a document comes from a sports topic, or it's about politics, or it's about entertainment, and so on. But as usual, we're going to start with a use case, which is an exciting one, around restaurant reviews.

So a classifier takes some input x, for example a sentence from the review or other inputs as we'll see. It pushes it through what's called a model to output some value y, that we're trying to predict. And here it's a class, for example, positive or negative. So positive in the case of sentiment analysis corresponds with thumbs up reviews, while negative corresponds with thumbs down. But this is just one example of classification.

You can look at text, for example, a web page, I want to figure out which webpages interest me, and I can align them to categories. For example, is this about education, a page about education, is it a page about finance, is it a page about technology and so on. So, there's not just two categories. There can be three, four, or even thousands of categories I'm predicting from



Now, another example of classification, which really has impacted all of our lives, is in spam filtering. Some of you might remember, perhaps in the early 2000s, what spam filters were like.

The quality was not very good. Really, they were all hand-tuned things where somebody said, oh it has certain words, it must be spam. But the spammers kept changing the words a little bit, adding numbers instead of letters, and beating the spam filters. And what really changed the world of spam filtering.

It changed it so much that I don't even look at my spam folder, actually, sorry if your message went to my spam folder, I just don't open it, is machine learning. It's classifiers. They took the input x of the email and they fed it through a classifier that predicted whether it's spam or not. It did that really well. And it does it by not just looking at the text of the email, but it looks at other characteristics.

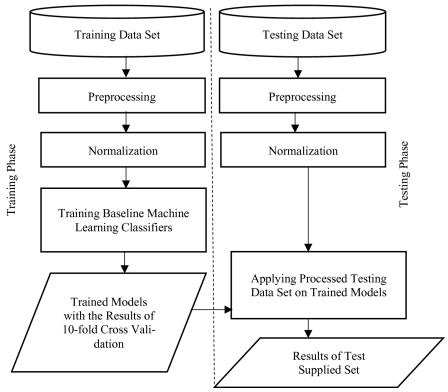
So for example, who sent it. If it somebody's whose a close friend, or somebody you have lots of vacation, with it's less likely to be spam. The IP address is a person sending from the usual computer and so on. Lots of information.

4.3.2 Linear Classifiers

n the field of machine learning, the goal of statistical classification is to use an object's characteristics to identify which class or group it belongs to.

A linear classifier achieves this by making a classification decision based on the value of a linear combination of the characteristics. An object's characteristics are also known as feature values and are typically presented to the machine in a vector called a feature vector. Such classifiers work well for practical problems such as document classification, and more generally for problems with many variables features, reaching accuracy levels comparable to non-linear classifiers while taking less time to train and use

There are two broad classes of methods for determining the parameters of a linear classifier. They can be generative and discriminative models.Methods of the first class model conditional density functions



Now when you have not just three non-zero words but in re-application you're gonna have tens of thousands of words with non-zero weight. And in that case we'll call those hyperplanes, really high dimensional separators called hyperplanes. Now, of course you can use more than linear classifiers. You can use more complex classifiers. And those, instead of having lines or hyperplanes, they have more complicated shapes or squiggly separations.

4.3.3 Evaluating Classification Models

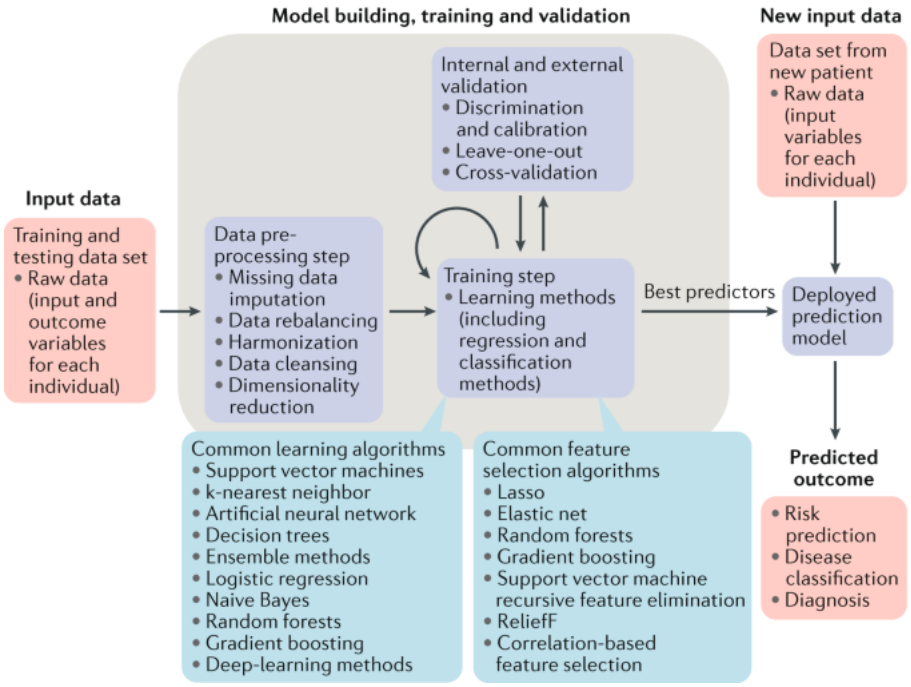
We talked about accuracy and errors that a classifier might make. But there are different kinds of errors. So this kind of errors are called types of mistakes. It's important to look at the types of mistakes a classifier might make. And one way to do that is through what's called a confusion matrix.

So, let's take that in a little bit. So we're talking about the relationship between the true label and whatever classifier predicts, so the predicted label. So let's say that if the true label is positive, and we predict a positive value for that sentence, we call that a true positive because we got it right. Similarly if the true label is negative and we predict that negative we call that a true negative.

So let's look at two applications, and what the cost has of false positives versus false negatives. So, if you consider spam filtering, a false negative is an email that was spam but went into my folder it thought it was not spam. So that's just annoying I got another spam email in my inbox.

Maybe it's bad but not super bad. However, if you look at a false positive that's an email that was not spam that got labeled as spam, went to my spam filter. I never saw it, I lost that email forever. That has a higher cost. Now we can also look at medical diagnosis or other applications as a second application.

So what's a false negative in medical diagnosis. False negative is, there's a disease that I have but it didn't get detected, so the classifier said it was negative. They don't have the disease. So in this case, the disease goes untreated, which can be a really bad thing.



But the false positives can also be a bad thing. That is, I classify as having the disease when I never had the disease. In this case I get treated potentially with a really bad drug or false side effect for diseases that I never had. So it's a little bit unclear what's worse, having a false positive or a false negative.

In medialc complications it really depends on the cost of the treatment and how many side effects it had versus how bad the disease can be. Now this relationship between the true label and the predicted label, false positive, false negatives, is called the Confusion Matrix. This matrix we just do. So for example, let's say that we have a setting with a 100 test examples.

4.4 Clustering And Retrieval Algorithms

In this week we have started to work on various clustering based algorithms and their applications using a case study. The following topics were discussed during the course of this week

1.Algorithms for retrieval

2.TF-IDF Vectors

2.Clustering models and algorithms

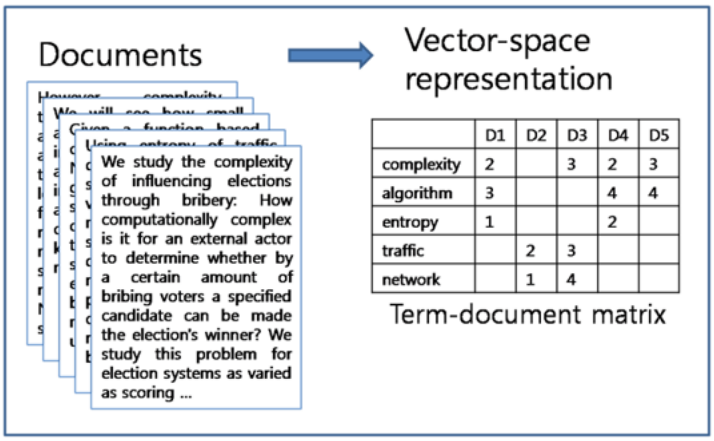
3.Summary of clustering

4.4.1 Algorithms for retrieval

There are lots and lots of articles out there and I can't expect him to go and read each of them and say yes, he's interested or no, he's not. So we like to think of a way to automatically retrieve a document that might be of interest to him. By questions here are first, how to we measure similarity between articles.

We need to have that in order to say that this article is similar to the one he's reading now and might also be of interest to him. Or that, here's a large set of articles that are very different and probably are not of interest to him. And then the second question is, how are we gonna search over the articles that exist out there and retrieve the next article to recommend.

Okay, before we talk about important words, let's first talk about rare words. So again, let's imagine that we're reading an article about soccer, and in this article there are lots and lots of common words like the and player and field and goal. And what we mean by a common word, it's a word that appears frequently in the corpus. In the corpus, that's just terminology for all the documents out there that we are looking at when we're doing this task of retrieval.



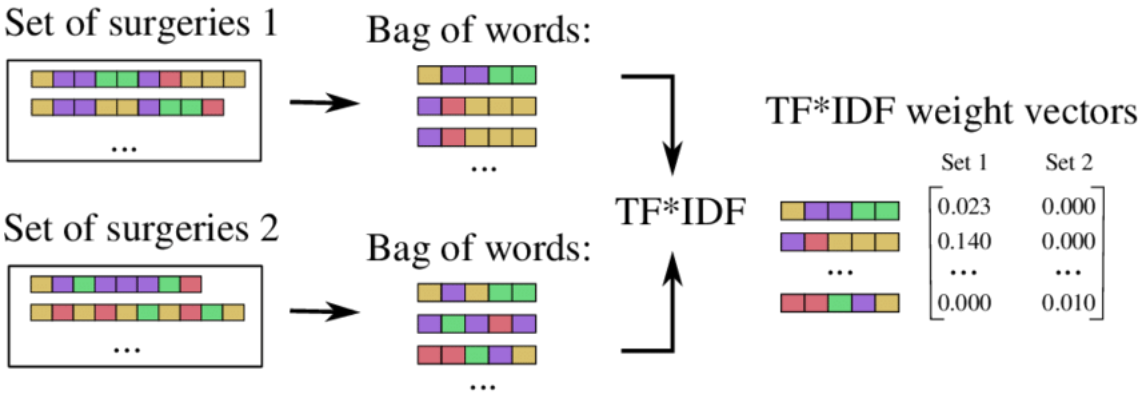
And what happens though is that these common words dominate the similarity metric that we talked about when we compare against other documents because, like I said, these words appear in lots and lots of documents. Whereas in contrast, there are some very rare words in this document we're looking at. Words like "futbol" and "Messi", the specific player we're reading about, that get completely swamped by all these common words.

4.4.2 TF-IDF Vectors

TF-IDF is an abbreviation for Term Frequency-Inverse Document Frequency and is a very common algorithm to transform text into a meaningful representation of numbers.

The technique is widely used to extract features across various NLP applications. To use a machine learning algorithm or a statistical technique on any form of text, it is prescribed to transform the text into some numeric or vector representation. This numeric representation should depict significant characteristics of the text.

There are many such techniques, for example, occurrence, term-frequency, TF-IDF, word co-occurrence matrix, word2vec and GloVe.F-IDF is an occurrence based numeric representation of text, let us understand the other primitive occurrence based techniques and how TF-IDF has evolved over them. One of the simplest ways to represent text in the form of numbers is how many times the word occurs in the entire corpus.



While computing term-frequency, each term is considered equally important and given a chance to participate in vector representation. But, there would be certain words which are so common across documents that they may contribute very little in deciding the meaning of it.

Term frequency of such words for example 'the', 'a', 'in', 'of' etc might suppress the weights of more meaningful words. Therefore, to reduce this effect, the term frequency is discounted by a factor called inverse document frequency.As a result, we have a vector representation which gives high value for a given term if that term occurs often in that particular document and very rarely anywhere else. If the term occurs in all the documents, idf computed would be 0. TF-IDF is the product of term-frequency and inverse document frequency.

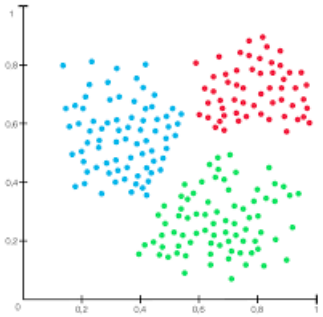
The inverse document frequency is a measure of how much information the word provides, i.e., if it's common or rare across all documents. It is the logarithmically scaled inverse fraction of the documents that contain the word (obtained by dividing the total number of documents by the number of documents containing the term, and then taking the logarithm of that quotient

4.4.3 Clustering Models And Algorithms

Clustering is a Machine Learning technique that involves the grouping of data points. Given a set of data points, we can use a clustering algorithm to classify each data point into a specific group.

In theory, data points that are in the same group should have similar properties and/or features, while data points in different groups should have highly dissimilar properties and/or features. Clustering is a method of unsupervised learning and is a common technique for statistical data analysis used in many fields.

In Data Science, we can use clustering analysis to gain some valuable insights from our data by seeing what groups the data points fall into when we apply a clustering algorithm.



To begin, we first select a number of classes/groups to use and randomly initialize their respective center points. To figure out the number of classes to use, it's good to take a quick look at the data and try to identify any distinct groupings. The center points are vectors of the same length as each data point vector and are the "X's" in the graphic above.Each data point is classified by computing the distance between that point and each group center, and then classifying the point to be in the group whose center is closest to it.

Based on these classified points, we recompute the group center by taking the mean of all the vectors in the group.Repeat these steps for a set number of iterations or until the group centers don't change much between iterations.

You can also opt to randomly initialize the group centers a few times, and then select the run that looks like it provided the best results.K-Means has the advantage that it's pretty fast, as all we're really doing is computing the distances between points and group centers; very few computations! It thus has a linear complexity O(n).

4.4.4 Summary Of Clustering

It is basically a type of unsupervised learning method . An unsupervised learning method is a method in which we draw references from datasets consisting of input data without labelled responses.

Generally, it is used as a process to find meaningful structure, explanatory underlying processes, generative features, and groupings inherent in a set of examples.

Clustering is the task of dividing the population or data points into a number of groups such that data points in the same groups are more similar to other data points in the same group and dissimilar to the data points in other groups. It is basically a collection of objects on the basis of similarity and dissimilarity between them.

Clustering is very much important as it determines the intrinsic grouping among the unlabeled data present. There are no criteria for a good clustering.

It depends on the user, what is the criteria they may use which satisfy their need. For instance, we could be interested in finding representatives for homogeneous groups (data reduction), in finding "natural clusters" and describe their unknown properties ("natural" data types), in finding useful and suitable groupings ("useful" data classes) or in finding unusual data objects (outlier detection). This algorithm must make some assumptions which constitute the similarity of points and each assumption make different and equally valid clusters.

Density-Based Methods : These methods consider the clusters as the dense region having some similarity and different from the lower dense region of the space. These methods have good accuracy and ability to merge two clusters.Example DBSCAN (Density-Based Spatial Clustering of Applications with Noise) , OPTICS (Ordering Points to Identify Clustering Structure) etc.

Hierarchical Based Methods : The clusters formed in this method forms a tree-type structure based on the hierarchy. New clusters are formed using the previously formed one.

Partitioning Methods : These methods partition the objects into k clusters and each partition forms one cluster. This method is used to optimize an objective criterion similarity function such as when the distance is a major parameter example K-means, CLARANS (Clustering Large Applications based upon Randomized Search.

Grid-based Methods : In this method the data space is formulated into a finite number of cells that form a grid-like structure. All the clustering operation done on these grids are fast and independent of the number of data objects example STING (Statistical Information Grid), wave cluster, CLIQUE (CLustering In Quest) etc.

4.5 Recommendation Systems

In this week we have started building recommender systems. Our case study involved studying a system which recommends songs. The complete breakdown of this week's learning is :

1.Recommender Systems

2.Co-occurence matrices

3.Matrix Factorization

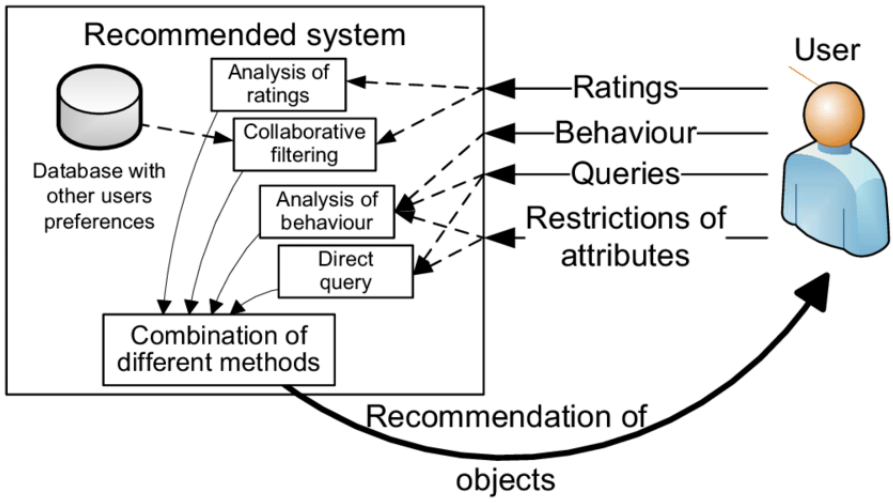
4.Performance Metrics

4.5.1 Recommender Systems

A recommender system, or a recommendation system, is a subclass of information filtering system that seeks to predict the ”rating” or ”preference” a user would give to an item.

They are primarily used in commercial applications.Recommender systems are utilized in a variety of areas and are most commonly recognized as playlist generators for video and music services like Netflix, YouTube and Spotify, product recommenders for services such as Amazon, or content recommenders for social media platforms such as Facebook and Twitter.

These systems can operate using a single input, like music, or multiple inputs within and across platforms like news, books, and search queries. There are also popular recommender systems for specific topics like restaurants and online dating. Recommender systems have also been developed to explore research articles and experts,collaborators, and financial services.



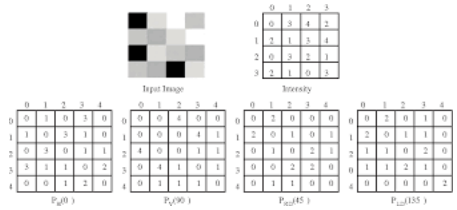
Recommender systems usually make use of either or both collaborative filtering and content-

based filtering, as well as other systems such as knowledge-based systems. Collaborative filtering approaches build a model from a user's past behavior as well as similar decisions made by other users.

This model is then used to predict items that the user may have an interest in. Content-based filtering approaches utilize a series of discrete, pre-tagged characteristics of an item in order to recommend additional items with similar properties. Current recommender systems typically combine one or more approaches into a hybrid system.

Each type of system has its strengths and weaknesses. In the above example, Last.fm requires a large amount of information about a user to make accurate recommendations. This is an example of the cold start problem, and is common in collaborative filtering systems.

Recommender systems are a useful alternative to search algorithms since they help users discover items they might not have found otherwise. Of note, recommender systems are often implemented using search engines indexing non-traditional data.



One approach to the design of recommender systems that has wide use is collaborative filtering.Collaborative filtering is based on the assumption that people who agreed in the past will agree in the future, and that they will like similar kinds of items as they liked in the past.

The system generates recommendations using only information about rating profiles for different users or items. By locating peer users/items with a rating history similar to the current user or item, they generate recommendations using this neighborhood. Collaborative filtering methods are classified as memory-based and model-based. A well-known example of memory-based approaches is the user-based algorithm,while that of model-based approaches is the Kernel-Mapping Recommender.

4.5.2 Co-occurance Matrices For Collaborative Filtering

Co-occurrence Filter is a boundary preserving filter. It is based on the Bilateral Filter but instead of using a Gaussian on the range values to preserve edges it relies on a co-occurrence matrix. Pixel values that co-occur frequently in the image will have a high weight in the co-occurrence matrix.

This, in turn, means that such pixel pairs will be averaged and hence smoothed, regardless of their intensity differences. On the other hand, pixel values that rarely co-occur will have a low weight in the co-occurrence matrix.

As a result, they will not be averaged and the boundary between them will be preserved. The CoF therefore extends the BF to deal with boundaries, not just edges. It learns co-occurrences directly from the image. We can achieve various filtering results by directing it to learn the co-occurrence matrix from a part of the image, or a different image. We give the definition of the filter, discuss how to use it with color images and show several use cases.

Whether considering the intensity or grayscale values of the image or various dimensions of color, the co-occurrence matrix can measure the texture of the image. Because co-occurrence matrices are typically large and sparse, various metrics of the matrix are often taken to get a more useful set of features. Features generated using this technique are usually called Haralick features, after Robert Haralick.

Texture analysis is often concerned with detecting aspects of an image that are rotationally invariant. To approximate this, the co-occurrence matrices corresponding to the same relation, but rotated at various regular angles, are often calculated and summed.Texture measures like the co-occurrence matrix, wavelet transforms, and model fitting have found application in medical image analysis in particular.

4.5.3 Matrix Factorization

Dense vector representations of words have proven to be useful for natural language tasks such as determining semantic similarity, parsing, and translation. Recently, work by Mikolov et al. and others has inspired an investigation into the construction of word vectors using stochastic gradient descent methods.

Models tend to fall into one of two categories: matrix factorization or sampling from a sliding window: Baroni et al. refers to these as count and predict methods, respectively.

Computation of Co-occurrence Matrix

Image matrix

0	0	1	1
0	0	1	1
0	2	2	2
2	2	3	3

Find the **number of co-occurrences** of pixel i to the neighboring pixel value j

i/j	0	1	2	3
0	#(0,0)	#(0,1)	#(0,2)	#(0,3)
1	#(1,0)	#(1,1)	#(1,2)	#(1,3)
2	#(2,0)	#(2,1)	#(2,2)	#(2,3)
3	#(3,0)	#(3,1)	#(3,2)	#(3,3)

Pixel values: 0,1,2,3. **So,** $N = 4$

So, *size* of CM = 4x4

$d = 1$

θ = horizontal (0°)

In this case, our data are our observed ratings. So those are the black squares. And our parameters are these user and movie topic factors. Okay, so what we're gonna do is we're gonna estimate each of these from our observe reading. So only using these black cells, we're gonna try and estimate these L and R vectors and resulting matrices.

Matrix factorization is a really, really powerful tool. And it's been proven useful on lots of different applications. But there's one limitation, and that's a problem that we talked about a little bit earlier of the cold-start problem where this model still can't handle this problem of what if we get a new user or a new movie.

So that's the case where we have no ratings either for a specific user or a specific movie. So that might be a new movie that arrives or a new user arrives. So this is a really important problem. And one that, for example, Netflix faces all the time. How do we make predictions for these users or movies?

4.5.4 Performance Metrics

Recommender systems are growing progressively more popular in online retail because of their ability to offer personalized experiences to unique users. Mean Average Precision at K is typically the metric of choice for evaluating the performance of a recommender systems. However, the use of additional diagnostic metrics and visualizations can offer deeper and sometimes surprising insights into a model's performance. So in this case, we could think about just counting how many items we guessed that they liked versus they did not like and

compare that to how many of those items they actually liked versus did not like.

But the issue here is actually multifold. One is the fact that we really care more about what the person liked than what they didn't like and often we're faced with very imbalanced classes. So, for example, there are lots and lots and lots of products out there, but typically a user's only gonna like a very small subset of them.

And so if we use this type of metric, we can get very good accuracy by just saying that the user won't like any of the items. So not recommending anything will get pretty good performance according to this metric. But another issue is something else which relates to the cost of making these different decisions.

So often we're gonna assume that the user has a limited attention span, and so we can only recommend a certain number of items for that user to look at. So there is a much larger cost if out of this very small set of items we're allowed to recommend to this person there's no liked item. That has a much higher cost than if we missed some of the user's liked items in this set of recommended products.

The system has to just recommend everything. If you recommend everything, you're guaranteed to recommend the products that I like. So in this case what would recall be? It would just be 1, because all 5 of the products I liked were recommended. So this is because there are 5 out of 5 products recommended. But what's the resulting precision of doing this. Well, it can actually be arbitrarily small. Because if you have tons and tons and tons of products out there, and if I like only a very, very small number of them.

4.6 Deep Learning And Neural Networks

This was the final week of the course. In this week we learnt about the much awaited Neural Netwroks, their applications and implementation techniques, and why neural nets are better than conventional machine learning algorithm. The detailed breakdown of this week is as follows

1.Neural Networks

2.Deep Learning And Deep Features

3.Computer Vision

4.Summary of Deep Learning

4.6.1 Neural Networks

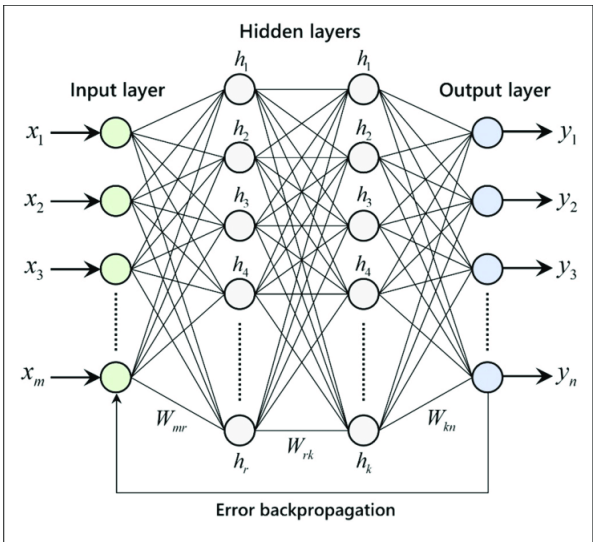
A neural network is a series of algorithms that endeavors to recognize underlying relationships in a set of data through a process that mimics the way the human brain operates.

In this sense, neural networks refer to systems of neurons, either organic or artificial in nature. Neural networks can adapt to changing input; so the network generates the best possible result without needing to redesign the output criteria. The concept of neural networks, which has its roots in artificial intelligence, is swiftly gaining popularity in the development of trading systems.

Neural networks, in the world of finance, assist in the development of such process as time-series forecasting, algorithmic trading, securities classification, credit risk modeling and constructing proprietary indicators and price derivatives. A neural network works similarly to the human brain's neural network.

A "neuron" in a neural network is a mathematical function that collects and classifies information according to a specific architecture. The network bears a strong resemblance to statistical methods such as curve fitting and regression analysis. A neural network contains layers of interconnected nodes.

Each node is a perceptron and is similar to a multiple linear regression. The perceptron feeds the signal produced by a multiple linear regression into an activation function that may be nonlinear.



n a multi-layered perceptron (MLP), perceptrons are arranged in interconnected layers. The input layer collects input patterns. The output layer has classifications or output signals to which input patterns may map.

For instance, the patterns may comprise a list of quantities for technical indicators about a security; potential outputs could be buy, hold or sell.

Hidden layers fine-tune the input weightings until the neural network's margin of error is minimal. It is hypothesized that hidden layers extrapolate salient features in the input data that have predictive power regarding the outputs. This describes feature extraction, which accomplishes a utility similar to statistical techniques such as principal component analysis.

Neural networks are broadly used, with applications for financial operations, enterprise planning, trading, business analytics and product maintenance.

Neural networks have also gained widespread adoption in business applications such as forecasting and marketing research solutions, fraud detection and risk assessment.A neural network evaluates price data and unearths opportunities for making trade decisions based on the data analysis.

The networks can distinguish subtle nonlinear interdependencies and patterns other methods of technical analysis cannot. According to research, the accuracy of neural networks in making price predictions for stocks differs. Some models predict the correct stock prices 50 to 60 percent of the time while others are accurate in 70 percent of all instances.1 Some have posited that a 10 percent improvement in efficiency is all an investor can ask for from a neural network.

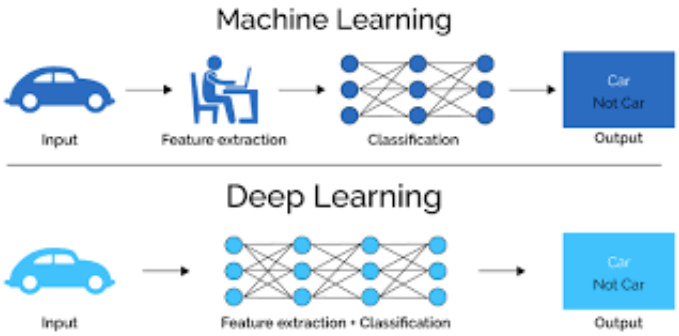
There will always be data sets and task classes that a better analyzed by using previously developed algorithms. It is not so much the algorithm that matters; it is the well-prepared input data on the targeted indicator that ultimately determines the level of success of a

neural network.

4.6.2 Deep Learning And Deep Features

Deep learning is an AI function that mimics the workings of the human brain in processing data for use in detecting objects, recognizing speech, translating languages, and making decisions.

Deep learning AI is able to learn without human supervision, drawing from data that is both unstructured and unlabeled. Deep learning, a form of machine learning, can be used to help detect fraud or money laundering, among other functions.



Deep learning has evolved hand-in-hand with the digital era, which has brought about an explosion of data in all forms and from every region of the world.

This data, known simply as big data, is drawn from sources like social media, internet search engines, e-commerce platforms, and online cinemas, among others. This enormous amount of data is readily accessible and can be shared through fintech applications like cloud computing.

However, the data, which normally is unstructured, is so vast that it could take decades for humans to comprehend it and extract relevant information. Companies realize the incredible potential that can result from unraveling this wealth of information and are increasingly adapting to AI systems for automated support.

One of the most common AI techniques used for processing big data is machine learning, a self-adaptive algorithm that gets increasingly better analysis and patterns with experience or with newly added data.

If a digital payments company wanted to detect the occurrence or potential for fraud in its system, it could employ machine learning tools for this purpose. The computational algorithm built into a computer model will process all transactions happening on the digital platform, find patterns in the data set, and point out any anomaly detected by the pattern.

Deep learning, a subset of machine learning, utilizes a hierarchical level of artificial neural networks to carry out the process of machine learning. The artificial neural networks are

built like the human brain, with neuron nodes connected together like a web. While traditional programs build analysis with data in a linear way, the hierarchical function of deep learning systems enables machines to process data with a nonlinear approach.

4.6.3 Computer Vision

Computer vision is an interdisciplinary scientific field that deals with how computers can gain high-level understanding from digital images or videos.

From the perspective of engineering, it seeks to understand and automate tasks that the human visual system can do.Computer vision tasks include methods for acquiring, processing, analyzing and understanding digital images, and extraction of high-dimensional data from the real world in order to produce numerical or symbolic information, e.g. in the forms of decisions.Understanding in this context means the transformation of visual images into descriptions of the world that make sense to thought processes and can elicit appropriate action.

This image understanding can be seen as the disentangling of symbolic information from image data using models constructed with the aid of geometry, physics, statistics, and learning theory.

The scientific discipline of computer vision is concerned with the theory behind artificial systems that extract information from images. The image data can take many forms, such as video sequences, views from multiple cameras, multi-dimensional data from a 3D scanner or medical scanning device. The technological discipline of computer vision seeks to apply its theories and models to the construction of computer vision systems.

There are many kinds of computer vision systems; however, all of them contain these basic elements: a power source, at least one image acquisition device, a processor, and control and communication cables or some kind of wireless interconnection mechanism. In addition, a practical vision system contains software, as well as a display in order to monitor the system.

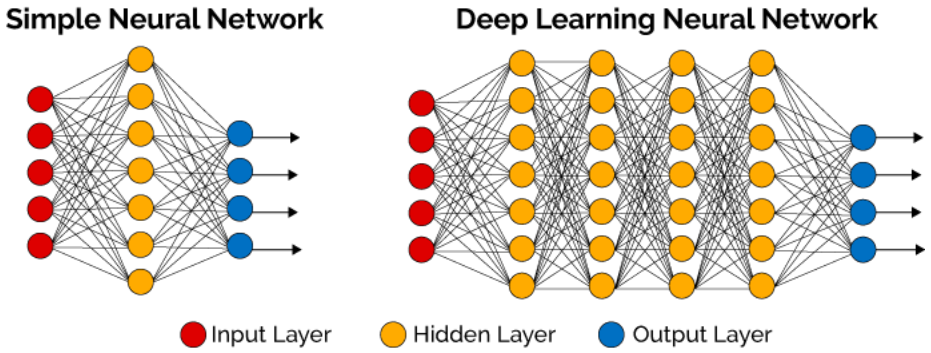
Vision systems for inner spaces, as most industrial ones, contain an illumination system and may be placed in a controlled environment. Furthermore, a completed system includes many accessories such as camera supports, cables and connectors. Most computer vision systems use visible-light cameras passively viewing a scene at frame rates of at most 60 frames per second

A few computer vision systems use image-acquisition hardware with active illumination or something other than visible light or both, such as structured-light 3D scanners, thermographic cameras, hyperspectral imagers, radar imaging, lidar scanners, magnetic resonance images, side-scan sonar, synthetic aperture sonar, etc. Such hardware captures images that are then processed often using the same computer vision algorithms used to process visible-light images.

4.6.4 Summary Of Deep Learning

Deep learning is part of a broader family of machine learning methods based on artificial neural networks with representation learning. Learning can be supervised, semi-supervised or unsupervised.

Deep learning architectures such as deep neural networks, deep belief networks, recurrent neural networks and convolutional neural networks have been applied to fields including computer vision, machine vision, speech recognition, natural language processing, audio recognition, social network filtering, machine translation, bioinformatics, drug design, medical image analysis, material inspection and board game programs, where they have produced results comparable to and in some cases surpassing human expert performance.



Most modern deep learning models are based on artificial neural networks, specifically, Convolutional Neural Networks (CNN)s, although they can also include propositional formulas or latent variables organized layer-wise in deep generative models such as the nodes in deep belief networks and deep Boltzmann machines.

In deep learning, each level learns to transform its input data into a slightly more abstract and composite representation. In an image recognition application, the raw input may be a matrix of pixels; the first representational layer may abstract the pixels and encode edges; the second layer may compose and encode arrangements of edges; the third layer may encode a nose and eyes; and the fourth layer may recognize that the image contains a face.

Importantly, a deep learning process can learn which features to optimally place in which level on its own.For supervised learning tasks, deep learning methods eliminate feature engineering, by translating the data into compact intermediate representations akin to principal components, and derive layered structures that remove redundancy in representation.

5 Assignments And Tasks

Every week we had an assignmnet to cement our understanding of the concepts. These assignmnets were all done in Python using the Jupyter Notebook. The details of each assignment is as follows

1. Predicting House Prices

2.Analyzing Sentiment

3.Document retrieval from wikipedia data

4.Song Recommender

5.Deep Features for Image Retrieval

5.1 Predicting House Prices

5.1.1 Objectives

In this assignment we want to load a data set of house prices in Our Jupyter notebook and then be able to manipulate that data to predict the houseprices in a certain region. We will be using Sframes, Jupyter notebook, Python environment and our datasets

5.1.2 Inputs

For this project we will be giving the following zip files as the input

1.HomeData.Sframe.zip

2.HouseImages.zip

5.1.3 Expected Outputs

After exporting all the necessary files we want to predict the prices of the home we want. The expected output is the prices of the house we want

5.1.4 Experimental Output

The experimental output is the same as the expected outuputs. We have created a simple linear regressor using the Turi-Create and we have tested it using our test data and then we predicted the house prices we wanted to predict.

Load house sales data

In [2]:
```
sales = turicreate.SFrame('~/data/home_data.sframe/')
```

In [3]:
```
sales
```

Out[3]:

id	date	price	bedrooms	bathrooms	sqft_living	sqft_lot	floors	waterfront
7129300520	2014-10-13 00:00:00+00:00	221900	3	1	1180	5650	1	0
6414100192	2014-12-09 00:00:00+00:00	538000	3	2.25	2570	7242	2	0
5631500400	2015-02-25 00:00:00+00:00	180000	2	1	770	10000	1	0
2487200875	2014-12-09 00:00:00+00:00	604000	4	3	1960	5000	1	0
1954400510	2015-02-18 00:00:00+00:00	510000	3	2	1680	8080	1	0
7237550310	2014-05-12 00:00:00+00:00	1225000	4	4.5	5420	101930	1	0
1321400060	2014-06-27 00:00:00+00:00	257500	3	2.25	1715	6819	2	0
2008000270	2015-01-15 00:00:00+00:00	291850	3	1.5	1060	9711	1	0
2414600126	2015-04-15 00:00:00+00:00	229500	3	1	1780	7470	1	0
3793500160	2015-03-12 00:00:00+00:00	323000	3	2.5	1890	6560	2	0

view	condition	grade	sqft_above	sqft_basement	yr_built	yr_renovated	zipcode	lat
0	3	7	1180	0	1955	0	98178	47.51123398

Simple regression model that predicts price from square feet

```
training_set, test_set = sales.random_split(.8,seed=0)
```

train simple regression model

```
sqft_model = turicreate.linear_regression.create(training_set,target='price',features=['sqft_living'])
```

```
PROGRESS: Creating a validation set from 5 percent of training data. This may take a while.
          You can set ``validation_set=None`` to disable validation tracking.


Linear regression:

--------------------------------------------------------

Number of examples          : 16466

Number of features          : 1

Number of unpacked features : 1

Number of coefficients    : 2

Starting Newton Method
```

Build a model with these additional features

```
my_features_model = turicreate.linear_regression.create(training_set,target='price',features=my_features)
```

```
PROGRESS: Creating a validation set from 5 percent of training data. This may take a while.
          You can set ``validation_set=None`` to disable validation tracking.


Linear regression:

--------------------------------------------------------

Number of examples          : 16444

Number of features          : 6

Number of unpacked features : 6

Number of coefficients    : 115
```

Apply learned models to make predictions

```
house1 = sales[sales['id']=='5309101200']
```

```
house1
```

id	date	price	bedrooms	bathrooms	sqft_living	sqft_lot	floors	waterfront
5309101200	2014-06-05 00:00:00+00:00	620000	4	2.25	2400	5350	1.5	0

view	condition	grade	sqft_above	sqft_basement	yr_built	yr_renovated	zipcode	lat
0	4	7	1460	940	1929	0	98117	47.67632376

long	sqft_living15	sqft_lot15
-122.37010126	1250.0	4880.0

5.2 Analyzing Sentiment

5.2.1 Objective

Our main objective for this project is to use classifiers and classify project sentiment. We read the product review data, and then we examine it. Then we proceed to build word count vectors of revies for our classfier and define what positive and negative reviews are. Then we train our classifier and predict and check the accuracy of the prediction

5.2.2 inputs

The following zip file was given as a input to the Notebook

1.AmazonBaby. Sframes.zip

5.2.3 Expected Outputs

After training the model we expected the model to give us the most positive and negative review of a certain product. We also wanted the prediction to have a high accuraccy.

5.2.4 Experimental Outputs

The actual output we got was similar to the one which we expected. We have trained our classifier successfully and then predicted the sentiment for a product named ”giraffee” and we also sorted the giraffee reviews according to the predicted sentiment

Analyze Product Sentiment

In [1]:
```
import turicreate
```

Read product review data

In [2]:
```
products = turicreate.SFrame('~/data/amazon_baby.sframe')
```

Define what is positive and negative sentiment

In [11]:
```
products['rating'].show()
```

```
Materializing SArray...

Done.

Materializing SArray...

Done.
```

Train our sentiment classifier

In [16]:
```
train_data,test_data = products.random_split(.8,seed=0)
```

In [17]:
```
sentiment_model = turicreate.logistic_classifier.create(train_data,target='sentiment', features=['word_count'], validation_set=te
```

```
WARNING: The number of feature dimensions in this problem is very large in comparison with the number of examples. Unless an ap
propriate regularization value is set, this model may not provide accurate predictions for a validation/test set.

Logistic regression:
```

Apply the sentiment classifier to better understand the Giraffe reviews

In [18]:
```
products['predicted_sentiment'] = sentiment_model.predict(products, output_type = 'probability')
```

In [19]:
```
products
```

Out[19]:

name	review	rating	word_count	sentiment
Planetwise Wipe Pouch	it came early and was not disappointed. i love ...	5.0	{'and': 3, 'love': 1, 'it': 2, 'highly': 1, ...	1
Annas Dream Full Quilt with 2 Shams ...	Very soft and comfortable and warmer than it ...	5.0	{'and': 2, 'quilt': 1, 'it': 1, 'comfortable': ...	1
Stop Pacifier Sucking without tears with ...	This is a product well worth the purchase. I ...	5.0	{'ingenious': 1, 'and': 3, 'love': 2, ...	1
Stop Pacifier Sucking without tears with ...	All of my kids have cried non-stop when I tried to ...	5.0	{'and': 2, 'parents!!': 1, 'all': 2, 'puppet.': ...	1
Stop Pacifier Sucking without tears with ...	When the Binky Fairy came to our house, we didn't ...	5.0	{'and': 2, 'this': 2, 'her': 1, 'help': 2, ...	1
A Tale of Baby's Days with Peter Rabbit ...	Lovely book, it's bound tightly so you may no ...	4.0	{'shop': 1, 'noble': 1, 'is': 1, 'it': 1, 'as': ...	1
Baby Tracker® - Daily Childcare Journal, ...	Perfect for new parents. We were able to keep ...	5.0	{'and': 2, 'all': 1, 'right': 1, 'when': 1, ...	1
Baby Tracker® - Daily Childcare Journal, ...	A friend of mine pinned this product on Pinte ...	5.0	{'and': 1, 'help': 1, 'give': 1, 'is': 1, ' ...	1

Sort the Giraffe reviews according to predicted sentiment

In [22]:
```
giraffe_reviews = giraffe_reviews.sort('predicted_sentiment', ascending=False)
```

In [23]:
```
giraffe_reviews
```

Out[23]:

name	review	rating	word_count	sentiment
Vulli Sophie the Giraffe Teether ...	Sophie, oh Sophie, your time has come. My ...	5.0	{'giggles': 1, 'all': 1, "violet's": 2, 'bring': ...	1
Vulli Sophie the Giraffe Teether ...	I'm not sure why Sophie is such a hit with the ...	4.0	{'adoring': 1, 'find': 1, 'month': 1, 'bright': 1, ...	1
Vulli Sophie the Giraffe Teether ...	I'll be honest...I bought this toy because all the ...	4.0	{'all': 2, 'discovered': 1, 'existence.': 1, ...	1
Vulli Sophie the Giraffe Teether ...	We got this little giraffe as a gift from a ...	5.0	{'all': 2, "don't": 1, '(literally).so': 1, ...	1
Vulli Sophie the Giraffe Teether ...	As a mother of 16month old twins; I bought ...	5.0	{'cute': 1, 'all': 1, 'reviews.': 2, 'just' ...	1
Vulli Sophie the Giraffe Teether ...	Sophie the Giraffe is the perfect teething toy. ...	5.0	{'just': 2, 'both': 1, 'month': 1, 'ears,': 1, ...	1
Vulli Sophie the Giraffe Teether ...	Sophie la giraffe is absolutely the best toy ...	5.0	{'and': 5, 'the': 1, 'all': 1, 'that': 2, ...	1
Vulli Sophie the Giraffe Teether ...	My 5-mos old son took to this immediately. The ...	5.0	{'just': 1, 'shape': 2, 'mutt': 1, '"dog': 1, ...	1

Show the most positive reviews

In [25]:
```
giraffe_reviews[0]['review']
```

Out[25]:
```
"Sophie, oh Sophie, your time has come. My granddaughter, Violet is 5 months old and starting to teeth. What joy little Sophie
brings to Violet. Sophie is made of a very pliable rubber that is sturdy but not tough. It is quite easy for Violet to twist So
phie into unheard of positions to get Sophie into her mouth. The little nose and hooves fit perfectly into small mouths, and th
e drooling has purpose. The paint on Sophie is food quality.Sophie was born in 1961 in France. The maker had wondered why there
was nothing available for babies and made Sophie from the finest rubber, phthalate-free on St Sophie's Day, thus the name was b
orn. Since that time millions of Sophie's populate the world. She is soft and for babies little hands easy to grasp. Violet esp
ecially loves the bumpy head and horns of Sophie. Sophie has a long neck that easy to grasp and twist. She has lovely, sizable
spots that attract Violet's attention. Sophie has happy little squeaks that bring squeals of delight from Violet. She is able t
o make Sophie squeak and that brings much joy. Sophie's smooth skin is soothing to Violet's little gums. Sophie is 7 inches tal
l and is the exact correct size for babies to hold and love.As you well know the first thing babies grasp, goes into their mout
hs- how wonderful to have a toy that stimulates all of the senses and helps with the issue of teething. Sophie is small enough
to fit into any size pocket or bag. Sophie is the perfect find for babies from a few months to a year old. How wonderful to hea
r the giggles and laughs that emanate from babies who find Sophie irresistible. Viva La Sophie!Highly Recommended.  prisrob 12-
11-09"
```

In [26]:
```
giraffe_reviews[1]['review']
```

Out[26]:
```
"I'm not sure why Sophie is such a hit with the little ones, but my 7 month old baby girl is one of her adoring fans.  The rubb
er is softer and more pleasant to handle, and my daughter has enjoyed chewing on her legs and the nubs on her head even before
she started teething.  She also loves the squeak that Sophie makes when you squeeze her.  Not sure what it is but if Sophie is
amongst a pile of her other toys, my daughter will more often than not reach for Sophie.  And I have the peace of mind of knowi
ng that only edible and safe paints and materials have been used to make Sophie, as opposed to Bright Starts and other baby toy
```

5.3 Document Retrieval From Wikipedia Data

5.3.1 Objectives

In this project our main objective is to build TD-IDF vectors and to apply nearest neighbours for retrieval of articles. We test the TD-IDF vetors on various words in the wikipedi document.

5.3.2 Inputs

We import the following dataframe into the jupyter notebook:

1.PeopleWiki.Sframe.zip

5.3.3 Expected Outputs

The main output that we expect of our model is to retrieve information from the wikipedia article about the specific names we specify to the model. We expect the model to be fast and accurate

5.3.4 Experimental Output

The actual output of the project was similar to that of our expected output. We first took a look at the articles of various people like 'Obama' and 'George Cloney'. Then we have computed the TF-IDF vectors for the entire corpus of the article. Then by using the Nearest Neighbours classifer we were successfully able to retrieve the articles from the wikipedia page

Load some text data from Wikipedia

```
In [4]: people = turicreate.SFrame('~/data/people_wiki.sframe')

In [5]: people
```

Out[5]:

URI	name	text
<http://dbpedia.org/resource/Digby_Morrell> ...	Digby Morrell	digby morrell born 10 october 1979 is a former ...
<http://dbpedia.org/resource/Alfred_J._Lewy> ...	Alfred J. Lewy	alfred j lewy aka sandy lewy graduated from ...
<http://dbpedia.org/resource/Harpdog_Brown> ...	Harpdog Brown	harpdog brown is a singer and harmonica player who ...
<http://dbpedia.org/resource/Franz_Rottensteiner> ...	Franz Rottensteiner	franz rottensteiner born in waidmannsfeld lower ...
<http://dbpedia.org/resource/G-Enka> ...	G-Enka	henry krvits born 30 december 1974 in tallinn ...
<http://dbpedia.org/resource/Sam_Henderson> ...	Sam Henderson	sam henderson born october 18 1969 is an ...

Load some text data from Wikipedia

In [4]:
```
people = turicreate.SFrame('~/data/people_wiki.sframe')
```

In [5]:
```
people
```

Out[5]:

URI	name	text
<http://dbpedia.org/resource/Digby_Morrell> ...	Digby Morrell	digby morrell born 10 october 1979 is a former ...
<http://dbpedia.org/resource/Alfred_J._Lewy> ...	Alfred J. Lewy	alfred j lewy aka sandy lewy graduated from ...
<http://dbpedia.org/resource/Harpdog_Brown> ...	Harpdog Brown	harpdog brown is a singer and harmonica player who ...
<http://dbpedia.org/resource/Franz_Rottensteiner> ...	Franz Rottensteiner	franz rottensteiner born in waidmannsfeld lower ...
<http://dbpedia.org/resource/G-Enka> ...	G-Enka	henry krvits born 30 december 1974 in tallinn ...
<http://dbpedia.org/resource/Sam_Henderson> ...	Sam Henderson	sam henderson born october 18 1969 is an ...

Compute TF-IDF for the entire corpus of articles

In [17]:
```
people['word_count'] = turicreate.text_analytics.count_words(people['text'])
```

In [18]:
```
people
```

Out[18]:

URI	name	text	word_count
<http://dbpedia.org/resource/Digby_Morrell> ...	Digby Morrell	digby morrell born 10 october 1979 is a former ...	{'melbourne': 1, 'college': 1, 'parade': ...
<http://dbpedia.org/resource/Alfred_J._Lewy> ...	Alfred J. Lewy	alfred j lewy aka sandy lewy graduated from ...	{'time': 1, 'each': 1, 'rhythms': 1, '25hour': ...
<http://dbpedia.org/resource/Harpdog_Brown> ...	Harpdog Brown	harpdog brown is a singer and harmonica player who ...	{'time': 1, 'honored': 1, 'maple': 1, 'awarded' ...
<http://dbpedia.org/resource/Franz_Rottensteiner> ...	Franz Rottensteiner	franz rottensteiner born in waidmannsfeld lower ...	{'kurdlawitzpreis': 1, 'this': 1, 'occasion' ...
<http://dbpedia.org/resource/G-Enka> ...	G-Enka	henry krvits born 30 december 1974 in tallinn ...	{'curtis': 1, 'promo': 1, '2007': 1, 'cent': 1, ...
<http://dbpedia.org/resource/Sam_Henderson> ...	Sam Henderson	sam henderson born october 18 1969 is an ...	{'journal': 1, 'niblit': 1, 'hometown': 1, ...
<http://dbpedia.org/resource/Aaron_LaCrate> ...	Aaron LaCrate	aaron lacrate is an american music producer ...	{'including': 1, 'artists': 1, 'local' ...
<http://dbpedia.org/resource/Trevor_Ferguson> ...	Trevor Ferguson	trevor ferguson aka john farrow born 11 november ...	{'concordia': 1, 'creative': 1, 'centre': ...

Examine the TF-IDF for the Obama article

In [21]:
```
obama = people[people['name'] == 'Barack Obama']
obama[['tfidf']].stack('tfidf',new_column_name=['word','tfidf']).sort('tfidf',ascending=False)
```

Out[21]:

word	tfidf
obama	43.2956530720749
act	27.67822262297991
iraq	17.747378587965535
control	14.887060845181308
law	14.722935761763422
ordered	14.533373950913514
military	13.115932778499415
involvement	12.784385241175055
response	12.784385241175055
democratic	12.410688697332166

Apply nearest neighbors for retrieval of Wikipedia articles

Build the NN model

In [26]:
```
knn_model = turicreate.nearest_neighbors.create(people,features=['tfidf'],label='name')
```

```
<unknown>:36: DeprecationWarning: invalid escape sequence \c
```

```
Starting brute force nearest neighbors model training.
```

Use model for retrieval... for example, who is closest to Obama?

In [27]:
```
knn_model.query(obama)
```

```
Starting pairwise querying.
```

```
+--------------+---------+-------------+--------------+
| Query points | # Pairs | % Complete. | Elapsed Time |
+--------------+---------+-------------+--------------+
```

5.4 Song Recommender

5.4.1 Objective

The main objective of this project is to Create a song recommender. We also want to Create a very simple popularity recommender. Then we Use the popularity model to make some predictions. Then at last we Build a recommender with personalization.

5.4.2 Inputs

We import the following dataset to the notebook

1 SongData.Sframe.zip

5.4.3 Expected Output

In this project our expected output is our recommender system to recommend us songs based on certain personalized traits and preferences. We want to model to be quick and accurate

5.4.4 Experimental Output

The Experimental output reached all our expectations. We have successfully created a song recommender. Create a very simple popularity recommender. Then we Use the popularity model to make some predictions. Then at last we Build a recommender with personalization. The system was also able to recommend similar songs to us based on features

Build a song recommender system

In [1]:
```
import turicreate
```

Load some music data

In [2]:
```
song_data = turicreate.SFrame('~/data/song_data.sframe/')
```

Explore our data

In [3]:
```
song_data
```

Create a song recommender

In [7]:
```
train_data,test_data = song_data.random_split(.8,seed=0)
```

Create a very simple popularity recommender

In [8]:
```
popularity_model = turicreate.popularity_recommender.create(train_data,
                                                            user_id = 'user_id',
                                                            item_id = 'song')
```

```
Recsys training: model = popularity

Warning: Ignoring columns song_id, listen_count, title, artist;

    To use one of these as a target column, set target =

    and use a method that allows the use of a target.

Preparing data set.

    Data has 893580 observations with 66085 users and 9952 items.

    Data prepared in: 0.674589s

893580 observations to process; with 9952 unique items.
```

Use the popularity model to make some predictions

In [9]:
```
popularity_model.recommend(users=[users[0]])
```

Out[9]:

user_id	song	score	rank
279292bb36dbfc7f505e36ebf038c81eb1d1d63e ...	Sehr kosmisch - Harmonia	4754.0	1
279292bb36dbfc7f505e36ebf038c81eb1d1d63e ...	Undo - Björk	4227.0	2
279292bb36dbfc7f505e36ebf038c81eb1d1d63e ...	You're The One - Dwight Yoakam ...	3781.0	3
279292bb36dbfc7f505e36ebf038c81eb1d1d63e ...	Dog Days Are Over (Radio Edit) - Florence + The ...	3633.0	4
279292bb36dbfc7f505e36ebf038c81eb1d1d63e ...	Revelry - Kings Of Leon	3527.0	5
279292bb36dbfc7f505e36ebf038c81eb1d1d63e ...	Horn Concerto No. 4 in E flat K495: II. Romance ...	3161.0	6
279292bb36dbfc7f505e36ebf038c81eb1d1d63e ...	Secrets - OneRepublic	3148.0	7
279292bb36dbfc7f505e36ebf038c81eb1d1d63e ...	Hey_ Soul Sister - Train	2538.0	8
279292bb36dbfc7f505e36ebf038c81eb1d1d63e ...	Fireflies - Charttraxx Karaoke ...	2532.0	9

Build a recommender with personalization

In [11]:
```
personalized_model = turicreate.item_similarity_recommender.create(train_data,
                                                                   user_id = 'user_id',
                                                                   item_id = 'song')
```

```
Recsys training: model = item_similarity

Warning: Ignoring columns song_id, listen_count, title, artist;

    To use one of these as a target column, set target =

    and use a method that allows the use of a target.

Preparing data set.

    Data has 893580 observations with 66085 users and 9952 items.

    Data prepared in: 0.599646s

Training model from provided data.

Gathering per-item and per-user statistics.
```

Apply personalized model to make song recommendations

In [12]:
```
personalized_model.recommend(users=[users[0]])
```

Out[12]:

user_id	song	score	rank
279292bb36dbfc7f505e36ebf038c81eb1d1d63e ...	Riot In Cell Block Number Nine - Dr Feelgood ...	0.0374999940395	1
279292bb36dbfc7f505e36ebf038c81eb1d1d63e ...	Sei Lá Mangueira - Elizeth Cardoso ...	0.0331632643938	2
279292bb36dbfc7f505e36ebf038c81eb1d1d63e ...	The Stallion - Ween	0.0322580635548	3
279292bb36dbfc7f505e36ebf038c81eb1d1d63e ...	Rain - Subhumans	0.0314159244299	4
279292bb36dbfc7f505e36ebf038c81eb1d1d63e ...	West One (Shine On Me) - The Ruts ...	0.0306771993637	5
279292bb36dbfc7f505e36ebf038c81eb1d1d63e ...	Back Against The Wall - Cage The Elephant ...	0.0301204770803	6
279292bb36dbfc7f505e36ebf038c81eb1d1d63e ...	Life Less Frightening - Rise Against ...	0.0284431129694	7
279292bb36dbfc7f505e36ebf038c81eb1d1d63e ...	A Beggar On A Beach Of Gold - Mike And The ...	0.0230024904013	8
279292bb36dbfc7f505e36ebf038c81eb1d1d63e ...	Audience Of One - Rise Against ...	0.0193938463926	9
279292bb36dbfc7f505e36ebf	Blame It On The Boogie -	0.0189873427153	10

5.5 Deep Features For Image Retrieval

5.5.1 Objectives

The main objectives of this project to Load some CIFAR-10 images, Compute deep features from our images,Create a nearest neighbors model to retrieve images from deep features,Use image retrieval model with deep features to find similar images,Find images similar to a car,Create a lambda function to find and show nearest neighbors to an image.

5.5.2 Inputs

The following zip file was imported to the notebook

1.ImageTrainData.zip

2.ImageTestData.zip

5.5.3 Expected Output

The expected output is that our neural net shows us similar pictures to a car when we set the prediction to be a car. It is an image classification neural network.

5.5.4 Experimental Output

The actual output is similar to the expected output. The neural network was successfully able to classify various types of images

Image retrieval using deep features

In [ ]:
```
import turicreate
```

Load some CIFAR-10 images

In [ ]:
```
image_data = turicreate.SFrame('~/data/image_train_data/')
```

In [ ]:
```
image_data['image'].explore()
```

Compute deep features from our images

In [ ]:
```
#deep_learning_model = turicreate.load_model('imagenet_model_iter45')
#image_data['deep_features'] = deep_learning_model.extract_features(image_data)
```

In [ ]:
```
image_data
```

Create a nearest neighbors model to retrieve images from deep features

In [ ]:
```
knn_model = turicreate.nearest_neighbors.create(image_data,
                                                features = ['deep_features'],
                                                label = 'id')
```

Use image retrieval model with deep features to find similar images

In [ ]:
```
cat = image_data[18:19]
```

In [ ]:
```
cat['image'].explore()
```

In [ ]:
```
knn_model.query(cat)
```

In [ ]:
```
def get_images_from_ids(query_result):
    return image_data.filter_by(query_result['reference_label'],'id')
```

In [ ]:
```
cat_neighbors = get_images_from_ids(knn_model.query(cat))
```

In [ ]:
```
cat_neighbors['image'].explore()
```

Find images similar to a car

In [ ]:
```
car = image_data[8:9]
```

In [ ]:
```
car['image'].explore()
```

In [ ]:
```
get_images_from_ids(knn_model.query(car))['image'].explore()
```

Create a lambda function to find and show nearest neighbors to an image

In [ ]:
```
show_neighbors = lambda i: get_images_from_ids(knn_model.query(image_data[i:i+1]))['image'].explore()
```

In [ ]:
```
show_neighbors(8)
```

In [ ]:
```
show_neighbors(26)
```

In [ ]:
```
show_neighbors(500)
```

In [ ]:
```

```

6 PROJECT -Bitcoin Price Prediction

6.1 Objectives

The main objective of this project is to test my skills by working on a guided project. In this project I built my own Python program to predict the price of Bitcoin (BTC) using a machine learning technique called Support Vector Machine. So you can start trading and making money.

Bitcoin uses peer-to-peer technology to operate with no central authority or banks; managing transactions and the issuing of bitcoins is carried out collectively by the network.

Bitcoin is open-source; its design is public, nobody owns or controls Bitcoin and everyone can take part. Through many of its unique properties, Bitcoin allows exciting uses that could not be covered by any previous payment system.

It is a decentralized digital currency without a central bank or single administrator that can be sent from user to user on the peer-to-peer bitcoin network without the need for intermediaries.

Transactions are verified by network nodes through cryptography and recorded in a public distributed ledger called a blockchain. Bitcoins are created as a reward for a process known as mining. They can be exchanged for other currencies, products, and services.

6.2 Design

The design of the model is based on choosing a classfier than can predict the future values of bitcoin in the future with the highest accuraccy. For this purpose, I choose a Supoort Vector Machine Algorithm.

Support Vector Machines are effective in high dimensional spaces. It works well with clear margin of seperation. It is effective in cases where number of dimensions is greater than the number of samples.

6.3 Inputs

The following were used in the creation of the Support Vector Machine :

1. Pandas

2.Numpy

3. Dataset from blockchain Domain

The code will be written in Jupyter Notebook itself.

6.4 Expected Outputs

The expected outcome was that the program predicts the price of Bitcoin for the next 30 days. We also strived for atmost accuracy of the model

6.5 Experimental output

The model was successfully able to predict the prices of bitcoin for the next 30 days. The accuracy was over 74%. You can see the detiled process in the below attachments.

BITCOIN PRICE PREDICTOR

Importing The Libraries

In [ ]:
```
import numpy as np
import pandas as pd
```

Importing The Dataset

In [ ]:
```
from google.colab import files # Use to load data on Google Colab
uploaded = files.upload() # Use to load data on Google Colab
```

Store the data set into a variable. ¶

In [ ]:
```
df = pd.read_csv('BitcoinPrice.csv')
df.head(7)
```

	Date	Price
0	2018-08-25 00:00:00	6719.429231
1	2018-08-26 00:00:00	6673.274167
2	2018-08-27 00:00:00	6719.266154
3	2018-08-28 00:00:00	7000.040000
4	2018-08-29 00:00:00	7054.276429
5	2018-08-30 00:00:00	6932.662500
6	2018-08-31 00:00:00	6981.946154

Remove the date column from the dataframe.

In [ ]:
```
df.drop(['Date'], 1, inplace=True)
```

Show the first 7 rows of the new data set.

In [ ]:
```
df.head(7)
```

	Price
0	6719.429231
1	6673.274167
2	6719.266154
3	7000.040000
4	7054.276429
5	6932.662500
6	6981.946154

Create a variable for predicting 'n' days out into the future

In [ ]:
```
#A variable for predicting 'n' days out into the future
prediction_days = 30 #n = 30 days

#Create another column (the target or dependent variable) shifted 'n' units up
df['Prediction'] = df[['Price']].shift(-prediction_days)
```

Show the first 7 rows of the new data set.

In [ ]:
```
df.head(7)
```

	Price	Prediction
0	6719.429231	6639.304167
1	6673.274167	6412.459167
2	6719.266154	6468.631667
3	7000.040000	6535.476667
4	7054.276429	6677.342500
5	6932.662500	6550.474167
6	6981.946154	6593.135000

Show the last 7 rows of the new data set

In [ ]:
```
df.tail(7)
```

Show the last 7 rows of the new data set

In [ ]:
```
df.tail(7)
```

	Price	Prediction
358	10295.117500	NaN
359	10605.825833	NaN
360	10746.507692	NaN
361	10169.094167	NaN
362	10030.746667	NaN
363	10255.977500	NaN
364	10158.540833	NaN

CREATE THE INDEPENDENT DATA SET

In [ ]:
```
# Convert the dataframe to a numpy array and drop the prediction column
X = np.array(df.drop(['Prediction'],1))

#Remove the last 'n' rows where 'n' is the prediction_days
X= X[:len(df)-prediction_days]
print(X)
```

```
[[ 6719.42923077]
 [ 6673.27416667]
 [ 6719.26615385]
 [ 7000.04      ]
 [ 7054.27642857]
 [ 6932.6625    ]
 [ 6981.94615385]
 [ 7100.94666667]
 [ 7247.93538462]
 [ 7260.94923077]
 [ 7326.8525    ]
 [ 7113.06923077]
 [ 6433.27166667]
 [ 6444.80416667]
 [ 6366.1075    ]
 [ 6286.42583333]
```

CREATE THE INDEPENDENT DATA SET

In [ ]:
```
# Convert the dataframe to a numpy array and drop the prediction column
X = np.array(df.drop(['Prediction'],1))

#Remove the last 'n' rows where 'n' is the prediction_days
X= X[:len(df)-prediction_days]
print(X)
```

Split the data into 80% training and 20% testing data sets.

In [ ]:
```
# Split the data into 80% training and 20% testing
from sklearn.model_selection import train_test_split
x_train, x_test, y_train, y_test = train_test_split(X, y, test_size=0.2)
```

In [ ]:
```
# Set prediction_days_array equal to the last 30 rows of the original data set from the price column
prediction_days_array = np.array(df.drop(['Prediction'],1))[-prediction_days:]
print(prediction_days_array)
```

```
[[ 9774.2575    ]
 [ 9725.4025    ]
 [ 9500.32416667]
 [ 9533.97933333]
 [ 9539.7125    ]
 [ 9873.81166667]
 [10088.8       ]
 [10478.90166667]
 [10790.63      ]
 [10826.275     ]
 [11713.16166667]
 [11759.01916667]
 [11703.73833333]
 [11803.88833333]
 [11816.9125    ]
 [11586.1725    ]
 [11377.80416667]
 [11397.80166667]
 [11144.38916667]
 [10450.81333333]
 [ 9988.9475    ]
 [10230.73333333]
```

Create the Support Vector Regression model

In [ ]:
```
from sklearn.svm import SVR
# Create and train the Support Vector Machine
svr_rbf = SVR(kernel='rbf', C=1e3, gamma=0.00001)#Create the model
svr_rbf.fit(x_train, y_train) #Train the model
```

Test the models accuracy on the testing data sets.

In [ ]:
```
# Testing Model: Score returns the accuracy of the prediction.
# The best possible score is 1.0
svr_rbf_confidence = svr_rbf.score(x_test, y_test)
print("svr_rbf accuracy: ", svr_rbf_confidence)
```

```
svr_rbf accuracy:  0.7427602807178524
```

The model showing the accuracy score of about 74.27%

Print the models predicted values

In [ ]:
```
# Print the predicted value
svm_prediction = svr_rbf.predict(x_test)
print(svm_prediction)

print()

#Print the actual values
print(y_test)
```

```
[ 7475.52091701  7565.7937874   6371.2913099   4007.46275354
  3990.72339304  3996.70429613  5800.82400543 10365.58300965
  5993.69571569 10080.31146435  6936.79042765  3958.41092521
  7905.37798994  3878.6445779   9388.60930153 11416.95846172
 10623.38386077  5556.67212205 10555.36378435  3917.4565318
  3967.49452986 10525.15438859  7161.03319111  5643.08444743
  6238.58326017  4001.6501031  10582.9920167   8028.16810711
  3996.74772282  7584.78827755  3841.23638228 10536.75747326
  5602.48739155  4041.39060711  7680.13244139  3837.14702435
  3872.45552782  4023.66930964  8535.647841   11192.77461015
  3862.82594886  5848.70744603  3953.15542172  5526.65730491
  5723.42716202  5517.490656    3835.9818862   6982.14190021
 11029.01246906  4042.42478724 10307.97137884  4000.86074933
  9950.29204349 10159.30662724  5659.29705523  3876.86665116
  5875.74843683  5052.90664833  6103.85193588 10402.16444841
  4008.71058894  5579.28618325  4037.65486918  5760.70116054
  7178.91303882 10263.74087514  4579.82380463]
```

```
[ 5863.52333333  7765.68833333  3982.85083333  3813.88
  5305.275       5274.14583333  6465.9175     10092.75166667
  4263.78333333 10449.62666667  6568.54916667  3498.86833333
  6562.64166667  3998.4975      8033.30666667  9539.7125
  8163.66333333  6473.75333333 11545.63333333  3905.05833333
  5302.9575     11577.69538462  3278.37416667  4533.68083333
  6538.79        5527.80166667 11886.88615385  8543.03
  3632.395       8546.17166667  3863.33333333  9725.4025
  3435.34        5251.19        3392.405       3948.41166667
  3933.14333333  3995.32333333  6563.62833333 10021.325
  3623.07166667  3910.97333333  3589.26083333  6260.64583333
  6452.57166667  3858.34916667  3832.61307692  5697.92333333
  9160.0675      5126.83416667 11412.12416667  3567.29916667
 10292.38333333 10230.73333333  6618.56769231  3812.58583333
  6382.66833333  3604.1175      6309.45285714  9584.47583333
  3572.735       4293.84083333  5031.475       4015.60916667
  7914.53916667  9533.97933333  3599.84166667]
```

Now, print the model's Bitcoin price predictions for the next 30 days.

In [ ]:
```
svm_prediction = svr_rbf.predict(prediction_days_array)
print(svm_prediction)
```

```
[10307.03252124 10271.53152569 10173.42916952 10157.76394357
 10156.41376563 10297.59237834  9956.05024258 10325.25877779
  9967.21486498  9921.87854538 11046.34005016 10875.76592445
 11077.76843636 10681.48343849 10620.39533362 11364.70844677
 11342.90246537 11377.2749394  10574.66881963 10287.5663945
 10134.5019906   9896.14859173  9974.99273303  9979.69891582
 10315.9849669  10047.03802787  9880.8677999  10055.58556853
  9921.51819023  9885.01946928]
```

This succeessfully completes my internship training and project.