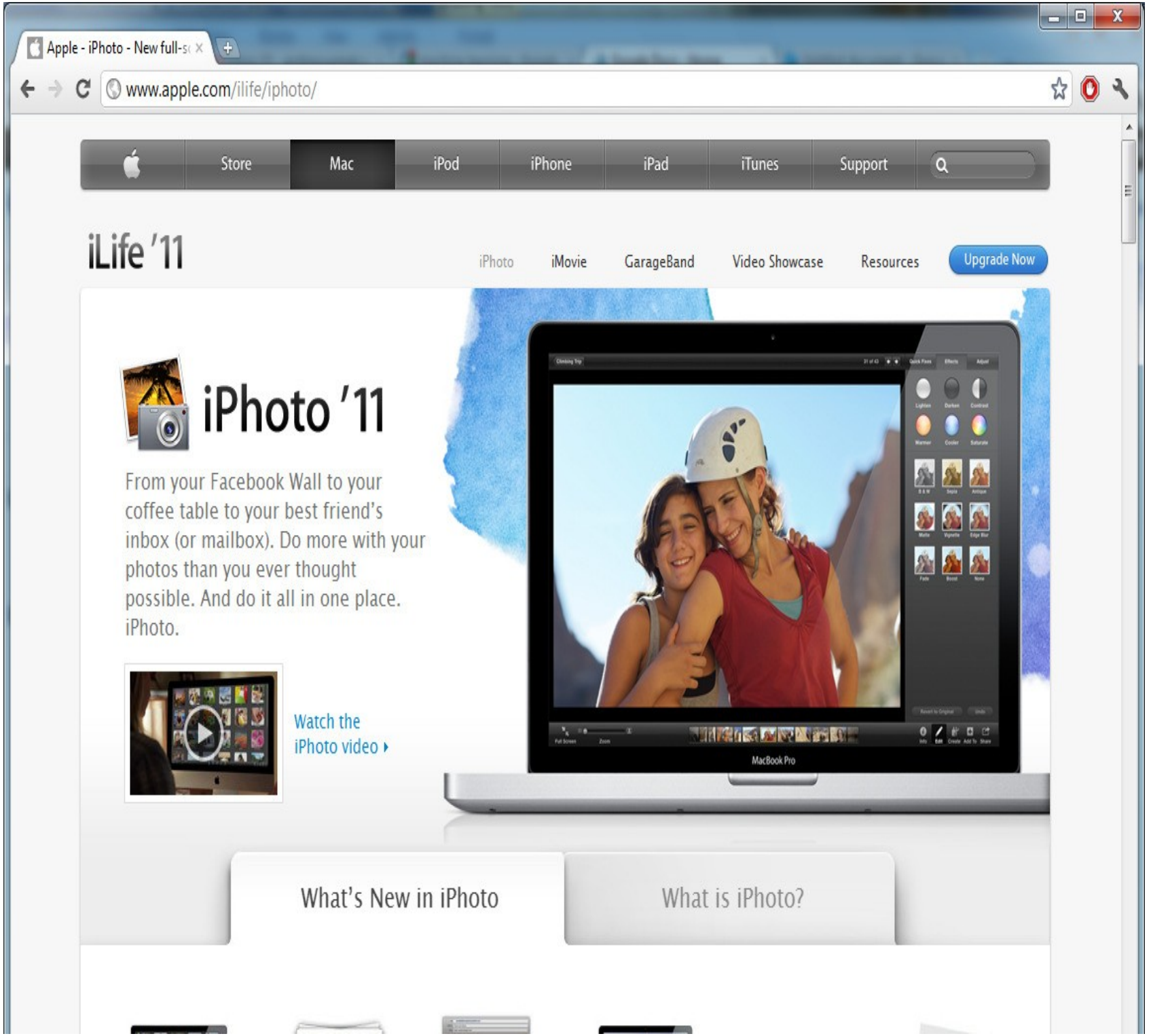
# Machine Learning Approach

# Introduction


Apple - iPhoto - New full-s
www.apple.com/ilife/iphoto/
Store
Mac
iPod
iPhone
iPad
iTunes
Support
iLife '11
iPhoto
iMovie
GarageBand
Video Showcase
Resources
Upgrade Now
iPhoto '11
From your Facebook Wall to your coffee table to your best friend's inbox (or mailbox). Do more with your photos than you ever thought possible. And do it all in one place. iPhoto.
Watch the iPhoto video
MacBook Pro
What's New in iPhoto
What is iPhoto?


SPAM

Machine Learning

- Grew out of work in AI
- New capability for computers

Examples:

- Database mining

  Large datasets from growth of automation/web.

  E.g., Web click data, medical records, biology, engineering
- Applications can’t program by hand.

  E.g., Autonomous helicopter, handwriting recognition, mos

  Natural Language Processing (NLP), Computer Vision.

Machine Learning

- Grew out of work in AI

- New capability for computers

Examples:

- Database mining

  Large datasets from growth of automation/web.

  E.g., Web click data, medical records, biology, engineering

- Applications can’t program by hand.
  E.g., Autonomous helicopter, handwriting recognition, most

  Natural Language Processing (NLP), Computer Vision.

- Self-customizing programs

  E.g., Amazon, Netflix product recommendations

Machine Learning

- Grew out of work in AI
- New capability for computers

Examples:

- Database mining

  Large datasets from growth of automation/web.

  E.g., Web click data, medical records, biology, engineering

- Applications can’t program by hand.

  E.g., Autonomous helicopter, handwriting recognition, most c

  Natural Language Processing (NLP), Computer Vision.

- Self-customizing programs

  E.g., Amazon, Netflix product recommendations

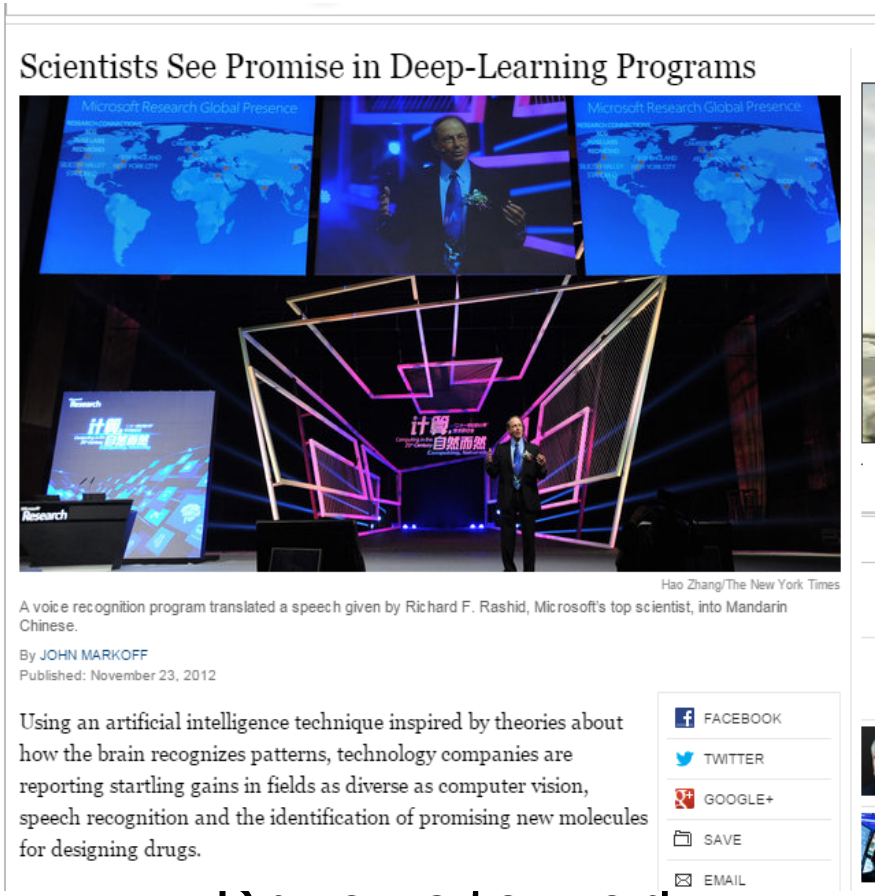
- Understanding human learning (brain, real AI).

# What do we want AI to do?

Keep us organized


Guide us to content


Help us find things



Scientists See Promise in Deep-Learning Programs

Hao Zhang/The New York Times

A voice recognition program translated a speech given by Richard F. Rashid, Microsoft's top scientist, into Mandarin Chinese.

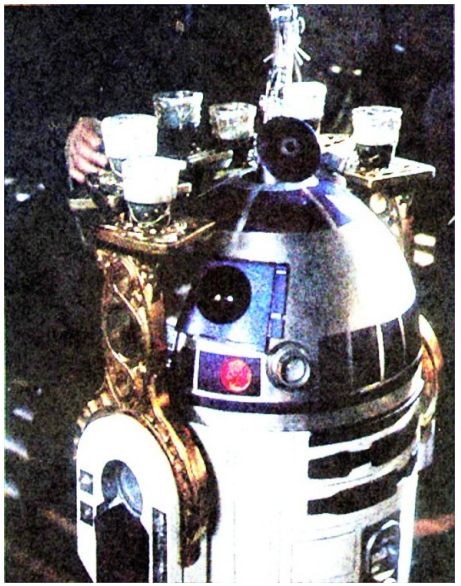
By JOHN MARKOFF
Published: November 23, 2012

Using an artificial intelligence technique inspired by theories about how the brain recognizes patterns, technology companies are reporting startling gains in fields as diverse as computer vision, speech recognition and the identification of promising new molecules for designing drugs.

FACEBOOK
TWITTER
GOOGLE+
SAVE
EMAIL

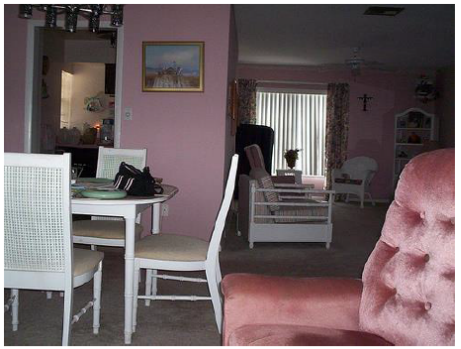
Help us communicate
帮助我们沟通

Drive us to work



Serve drinks?

Two dogs and one person.
A person is playing with a dog.



Six chairs and one table and one person.
The image is taken in a room.

A living room with a carpeting, a cream sofa and chair, and a large door.

The room in the apartment gets some afternoon sun.

Living room with white couch and blue carpeting.

Five bottles and one desk and one person.
A person is sitting on a chair

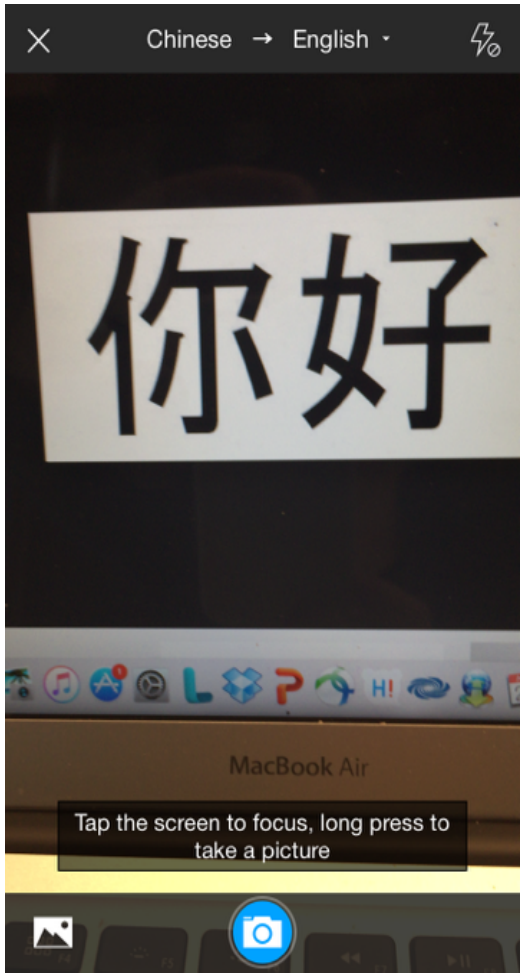
One motorbike and one person.
A person is riding a motorbike.

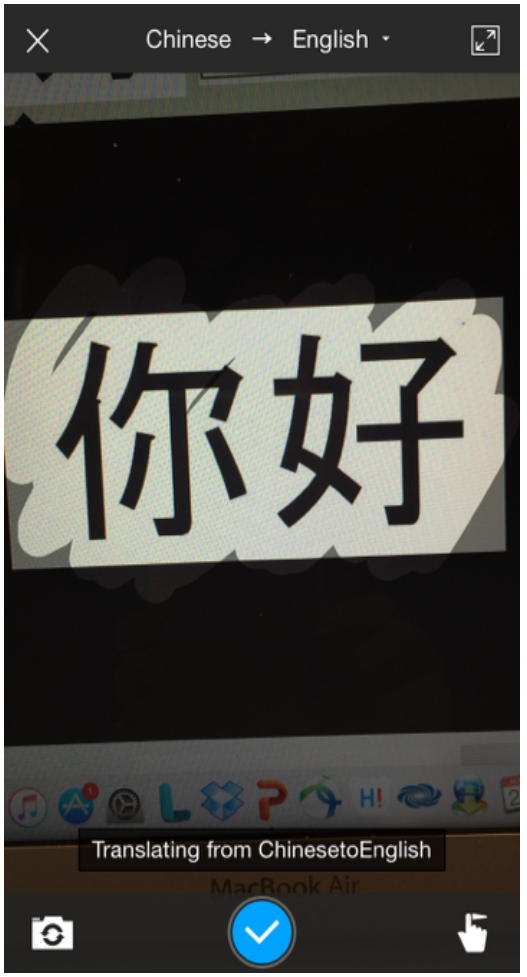
A yellow bus driving down a road with green trees and grass in the background.

School bus on a street.

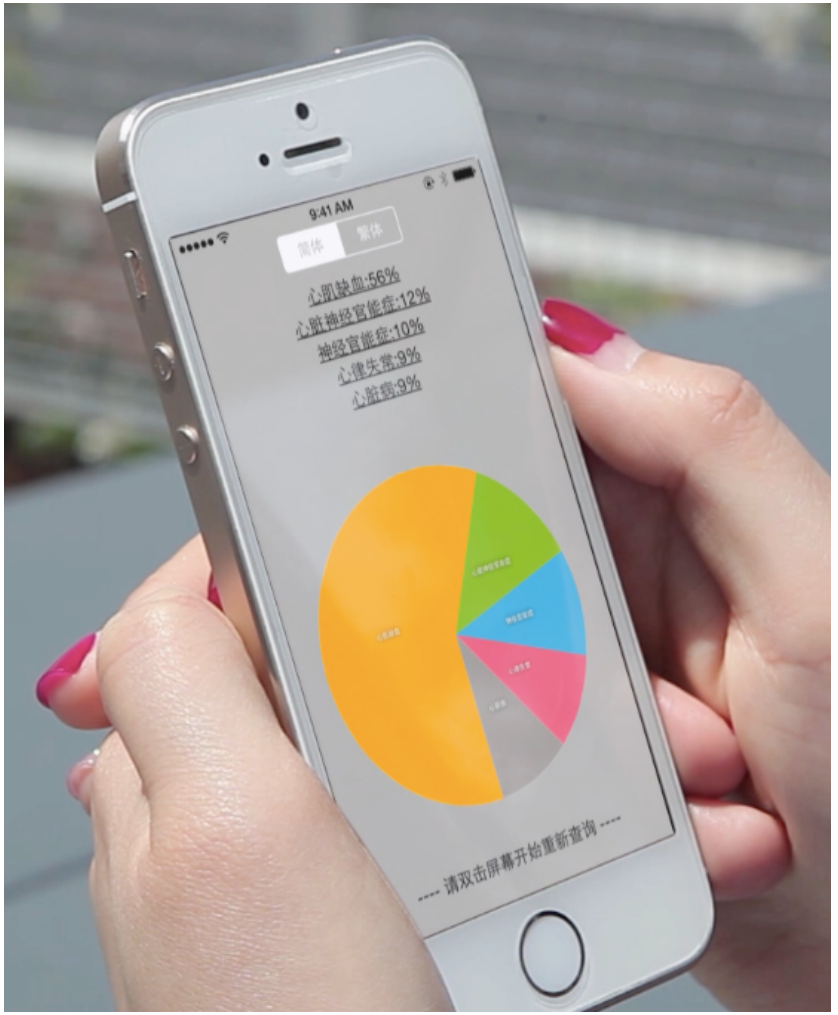
Yellow bus with trees in the background.

# OCR-based Translation App



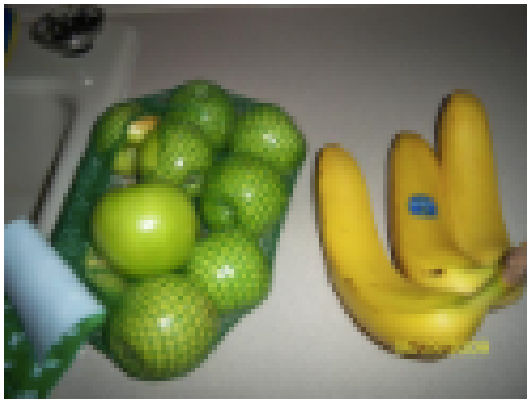
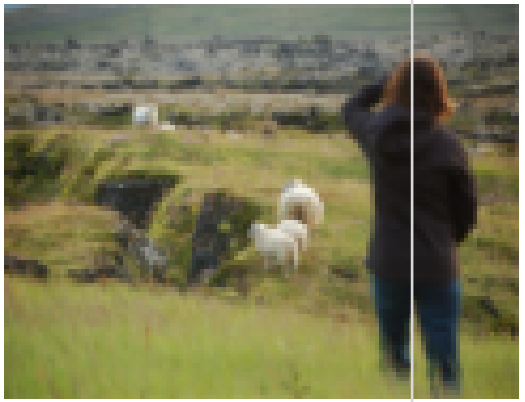




# Medical Diagnostics App

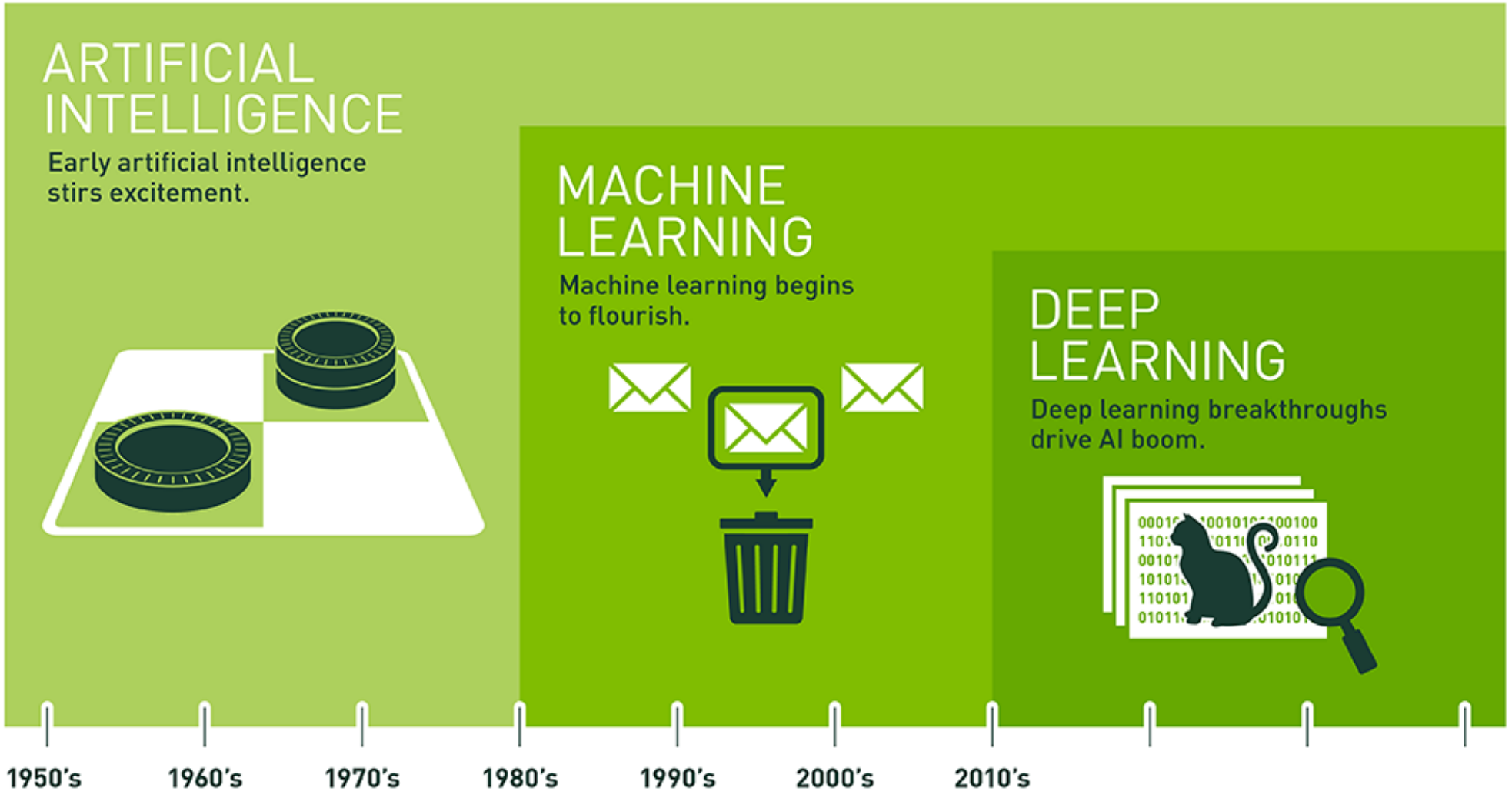



AskADoctor can assess 520 different diseases, representing ~90 percent of the most common medical problems.

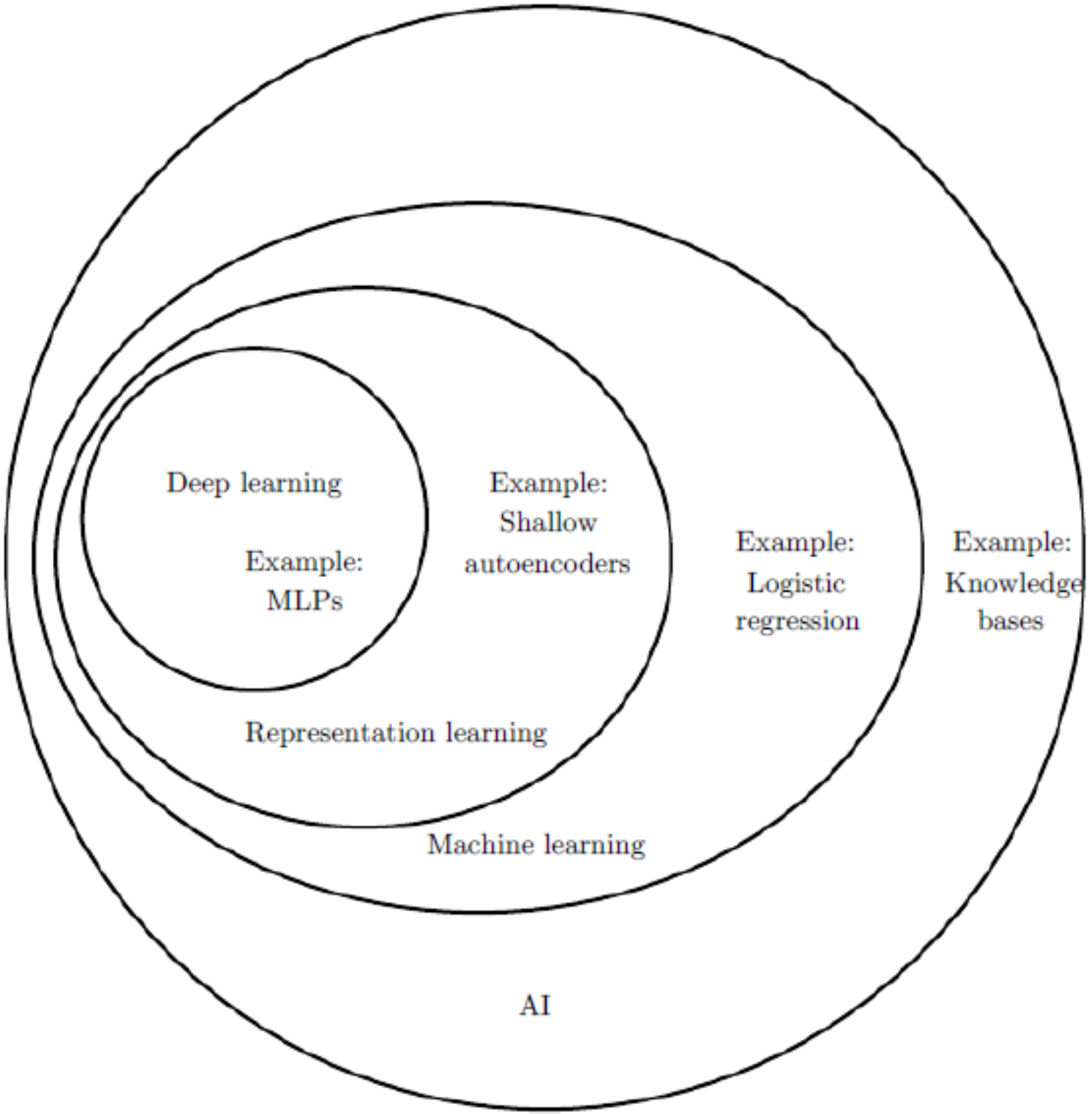
# Image Q&A

| | | | |
|---|---|---|---|
| Image |  |  |  |
| Question | 公共汽车是什么颜色的？<br>What is the color of the bus? | 黄色的是什么？<br>What is there in yellow? | 草地上除了人以外还有什么动<br>What is there on the grass, except<br>the person? |
| Answer | 公共汽车是红色的。<br>The bus is red. | 香蕉。<br>Bananas. | 羊。<br>Sheep. |

Sample questions and answers




Since an early flush of optimism in the 1950s, smaller subsets of artificial intelligence – first machine learning, then deep learning, a subset of machine learning – have created ever larger disruptions.


Deep learning
Example:
MLPs
Example:
Shallow
autoencoders
Representation learning
Example:
Logistic
regression
Machine learning
Example:
Knowledge
bases
AI

# Introduction

# What is machine learning

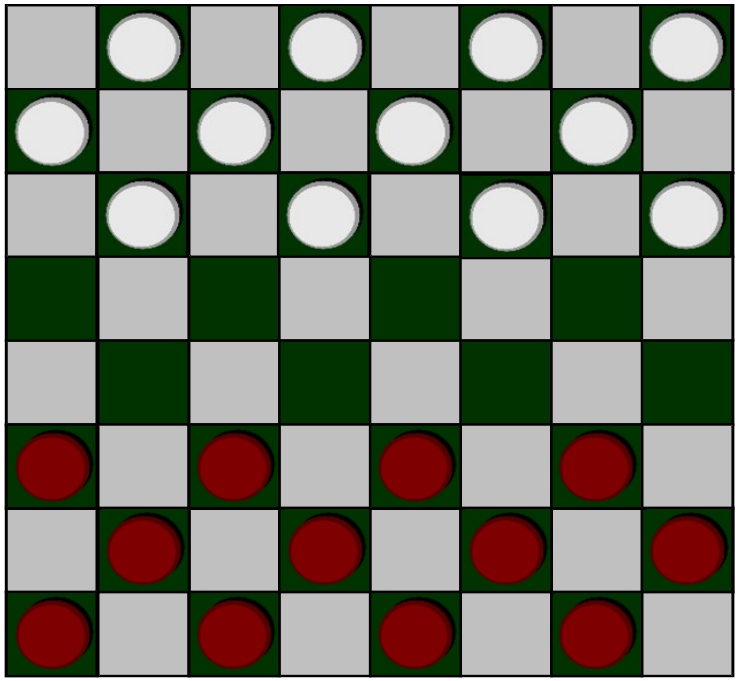
# Machine Learning definition

# Machine Learning definition

- Arthur Samuel (1959). Machine Learning: Field of study that gives computers the ability to learn without being explicitly programmed.

# Machine Learning definition

- Arthur Samuel (1959). Machine Learning: Field of study that gives computers the ability to learn without being explicitly programmed.



# Machine Learning definition

- Arthur Samuel (1959). Machine Learning: Field of study that gives computers the ability to learn without being explicitly programmed.
- Tom Mitchell (1998) Well-posed Learning Problem: A computer program is said to learn from experience E with respect to some task T and some performance measure P, if its performance on T, as measured by P, improves with experience E.

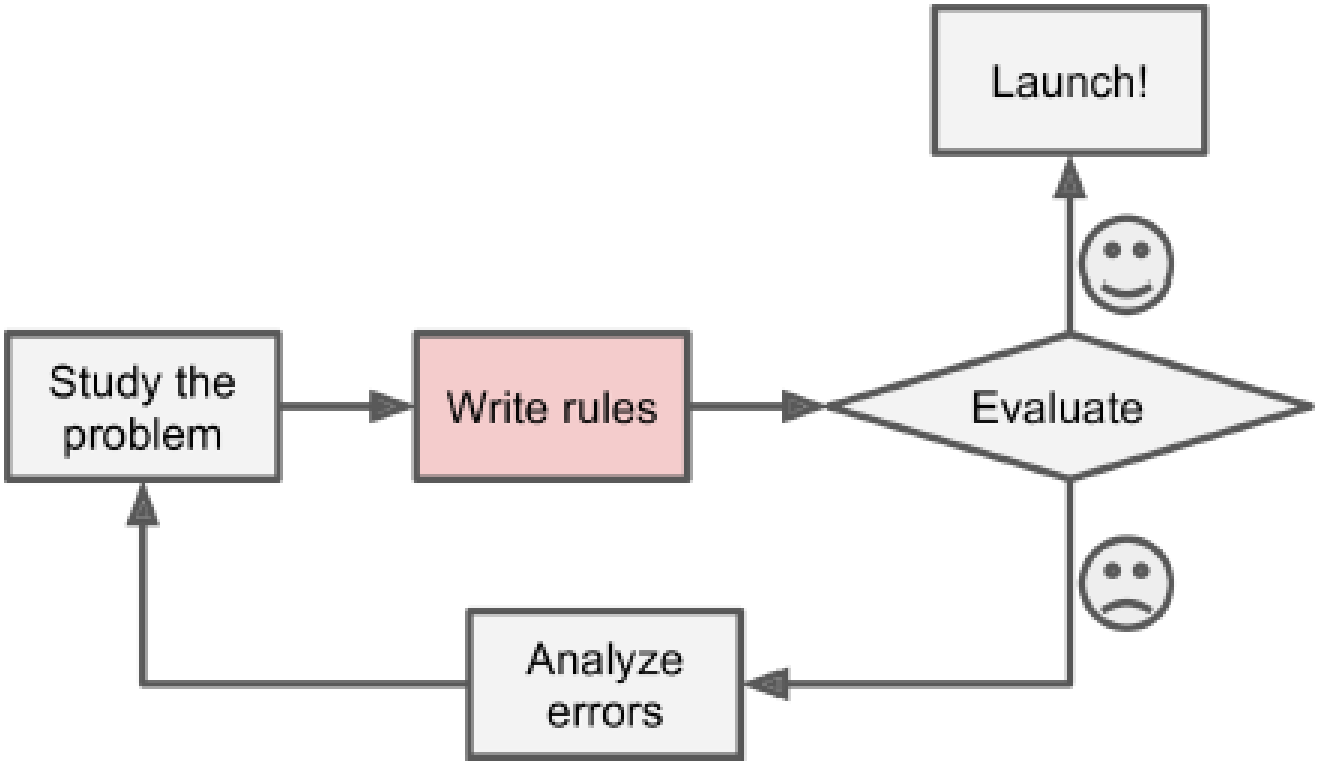
Machine learning algorithms:

- Supervised learning
- Unsupervised learning

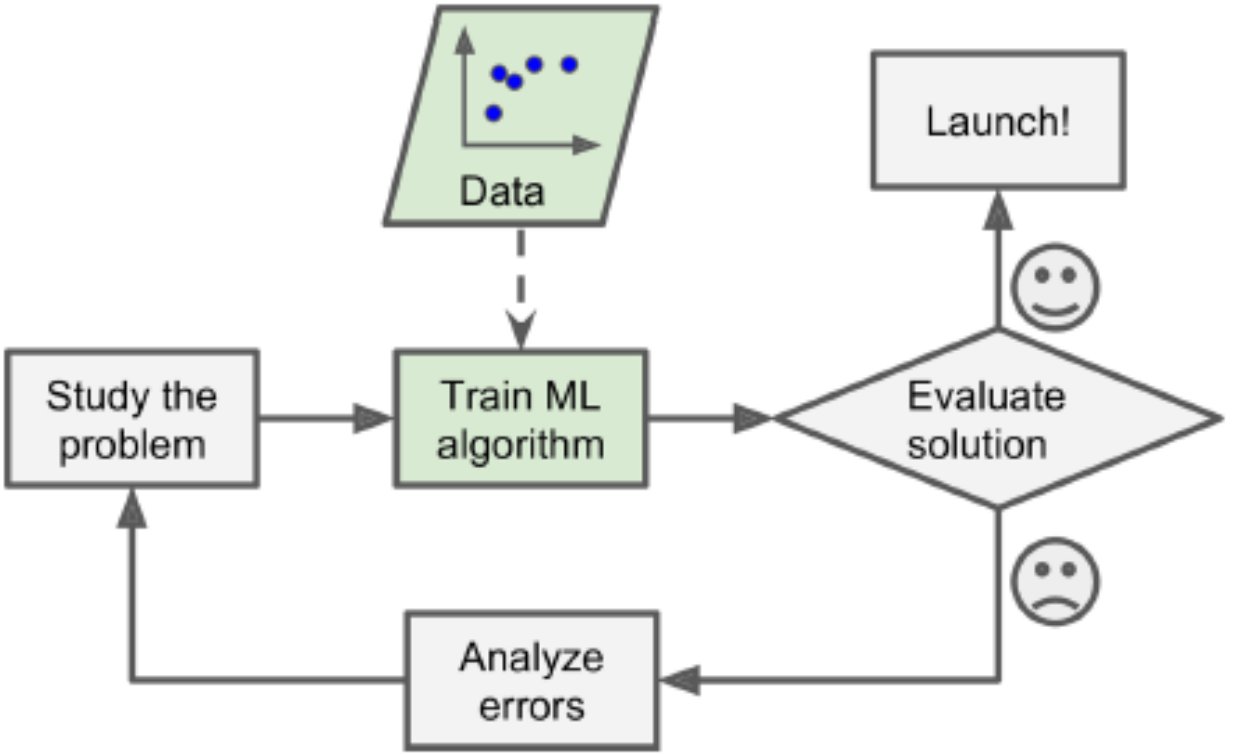
Others: Reinforcement learning, recommender systems.

Also talk about: Practical advice for applying learning algorithms.

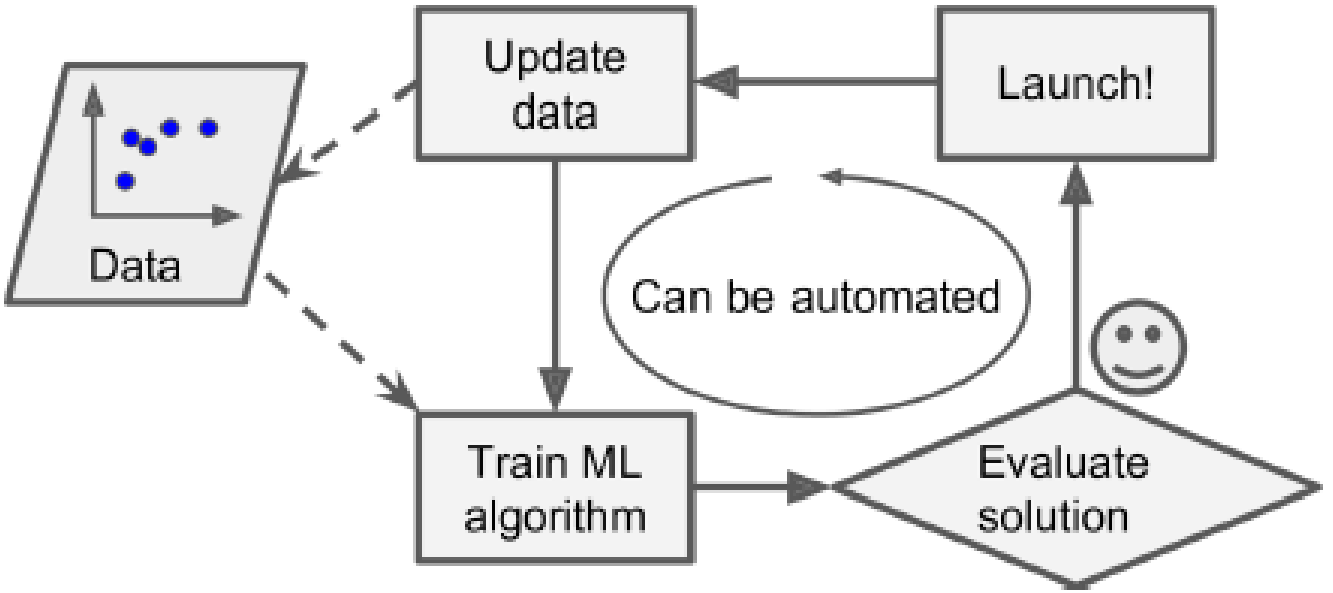
# Traditional Approach



# Machine Learning Appoach



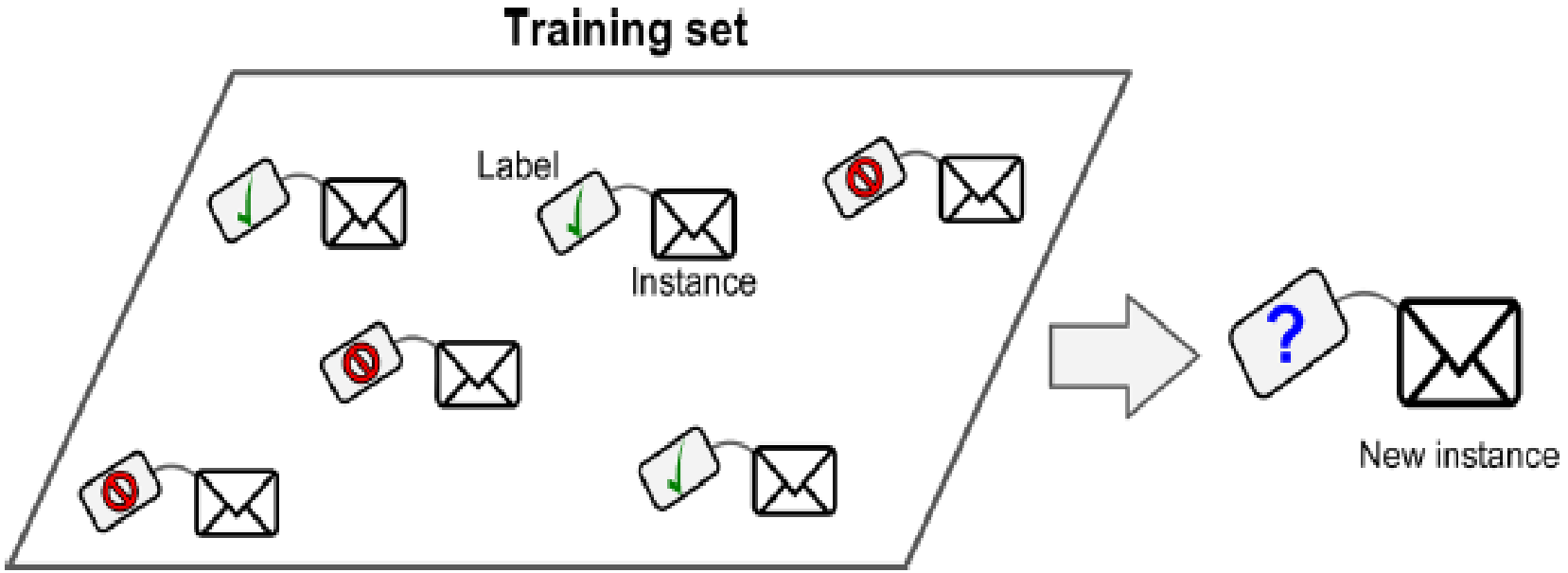
# Machine Learning Appoach:Adaptive



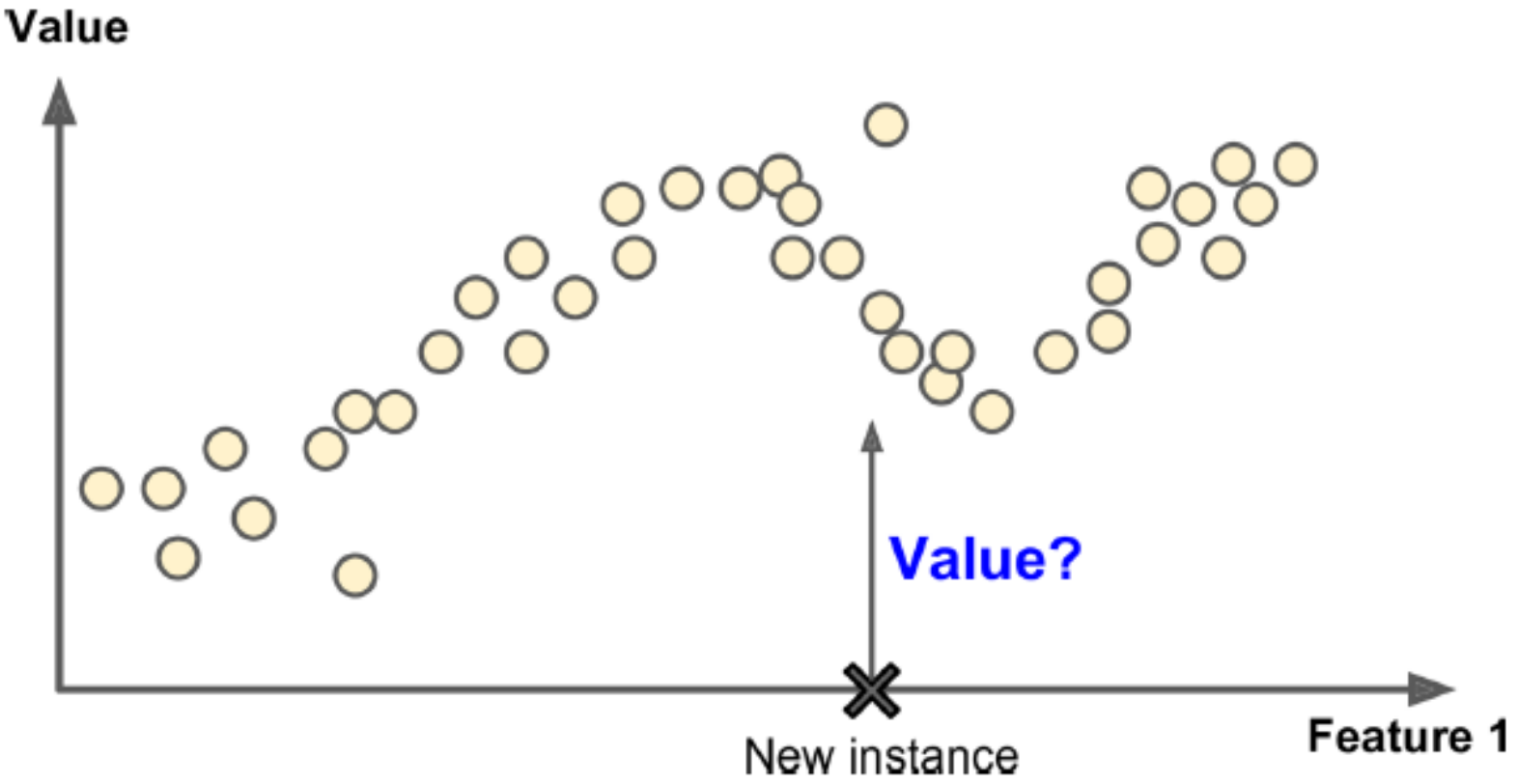
# Machine Learning Appoach

- Machine Learning is great for:
  - Problems for which existing solutions require a lot of hand-tuning or long lists of rules: one Machine Learning algorithm can often simplify code and perform better.
  - Complex problems for which there is no good solution at all using a traditiona approach: the best Machine Learning techniques can find a solution.
  - Fluctuating environments: a Machine Learning system can adapt to new data.
  - Getting insights about complex problems and large amounts of data.

# Machine Learning Appoach:Supervised



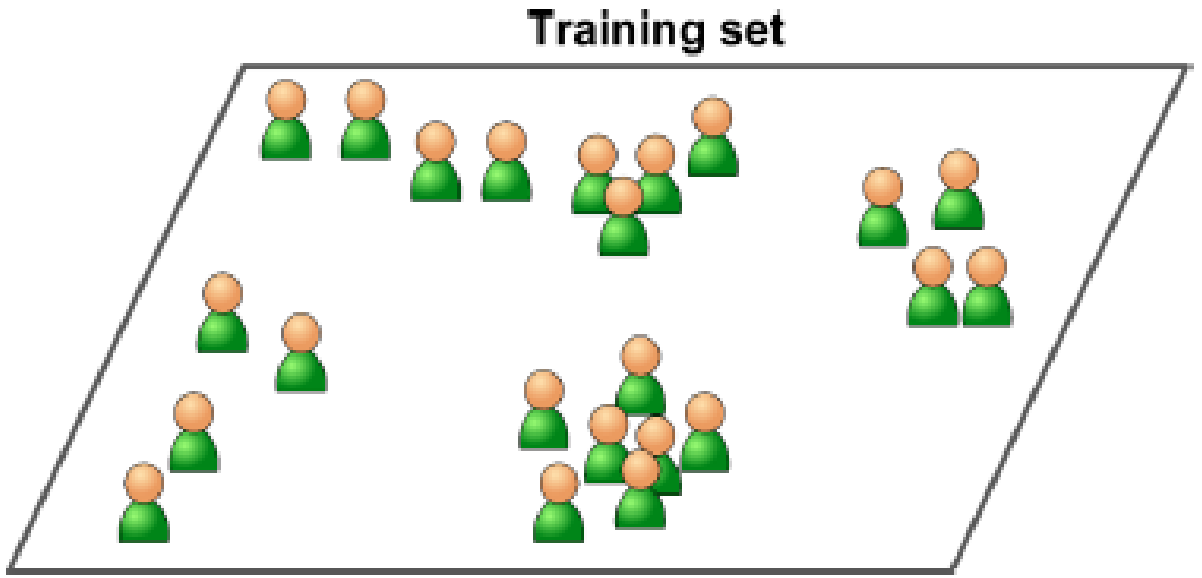
# Machine Learning Appoach:Supervised



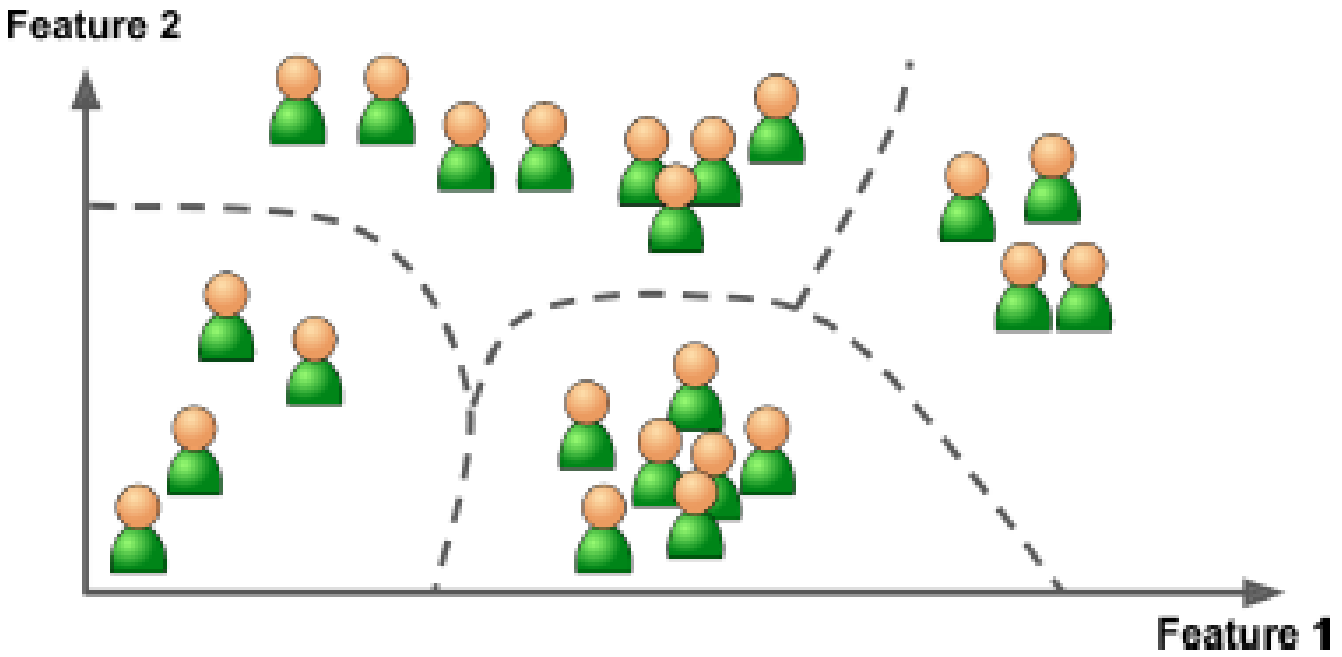
# Machine Learning Appoach:Supervised

- k-Nearest Neighbors
- Linear Regression
- Logistic Regression
- Support Vector Machines (SVMs)
- Decision Trees and Random Forests
- Neural networks

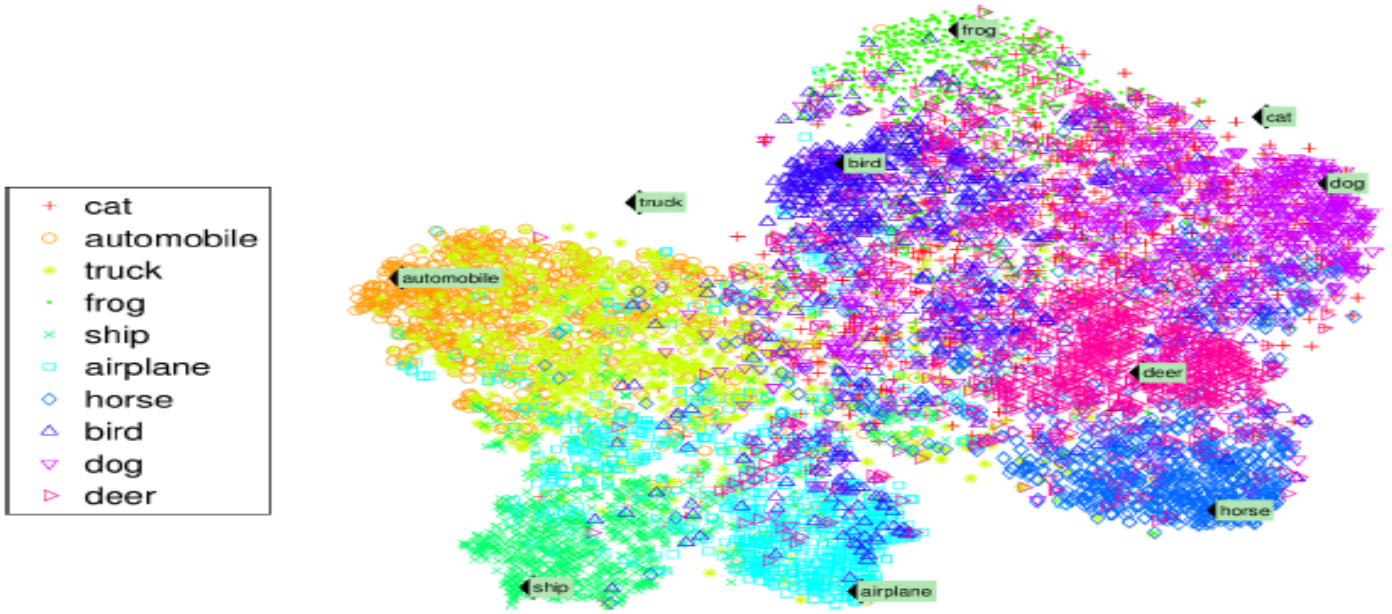
# Machine Learning Appoach:Unsupervised



# Machine Learning Appoach:Unsupervised:Clustering

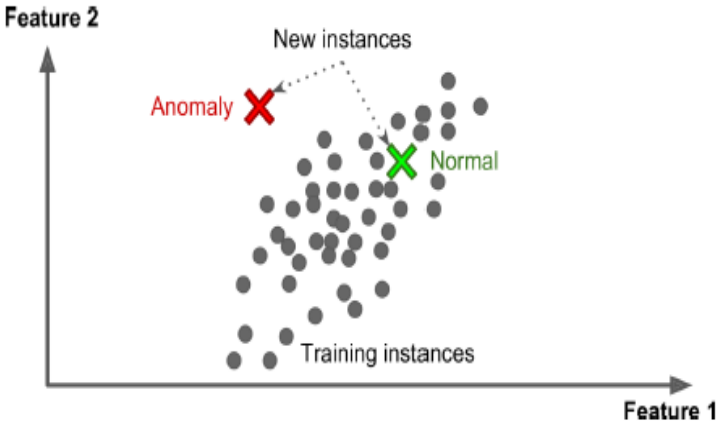


# Machine Learning Appoach:Unsupervised:Visualization




- Visualization algorithms are also good examples of unsupervised learning.
- Feed them lots of complex data and they output 2D or 3D representation that can easily be plotted.
- They try to preserve as much structure as possible.

# Machine Learning Appoach:Unsupervised:Anomaly Detection



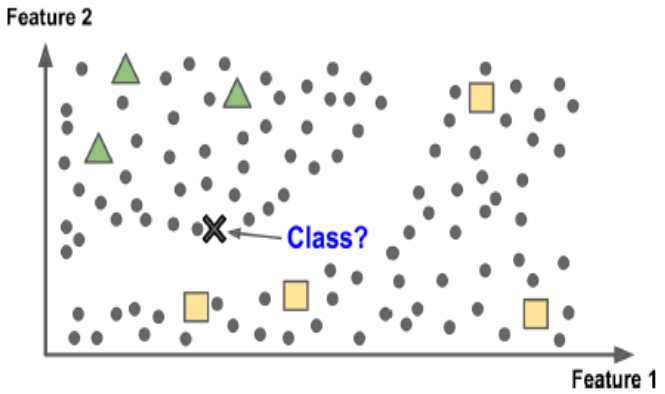

- Detecting unusual bank transactions
- Manufacturing defects
- Automatically removing outliers, before feeding to other ML algorithms.

- The system is trained on normal instances.
- When it sees a new one it can tell whether it is normal or an anomaly.

# Machine Learning Appoach:Unsupervised

- Clustering
  - k-Means
  - Hierarchical Cluster Analysis (HCA)
  - Expectation Maximization
- Visualization and dimensionality reduction
  - Principal Component Analysis (PCA)
  - Kernel PCA
  - Locally-Linear Embedding (LLE)
  - t-distributed Stochastic Neighbor Embedding (t-SNE)
- Association rule learning
  - Apriori
  - Eclat

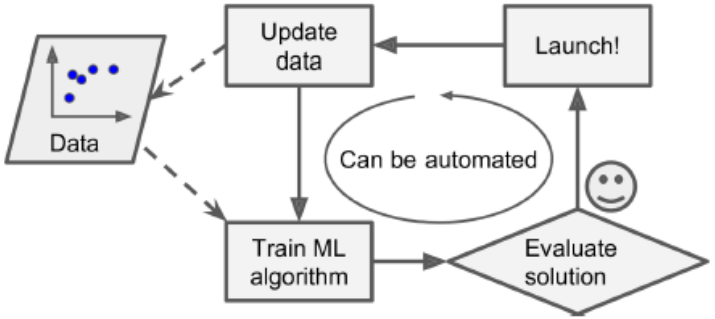
# Machine Learning Appoach:Semisupervised




- Deep Belief Networks (DBN)
- Restricted Boltzman machine (RBMs)
- RBMs are trained sequentially, unsupervised and fine tuned using supervised learning.

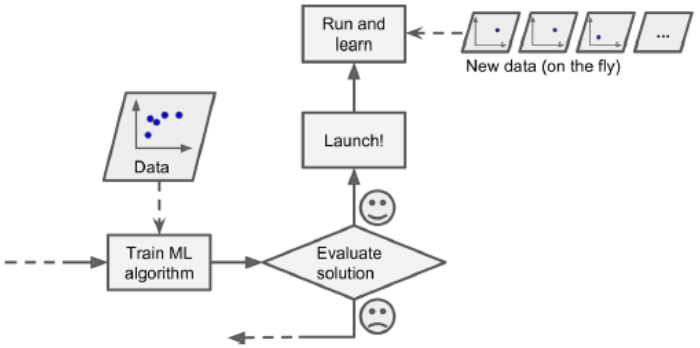
- google photos is a good example
  - Firstly photos are clustered
  - A appears in photos 01, 11, 12
  - B appears in " 02, 22, 23
- Now the system needs to tell it who these people are.
- One label/person.
- Most semi-supervised are a combination of supervised and unsupervised algorithms.

# Machine Learning Appoach:Batch




- incapable of learning incrementally
- takes a lot of time, done offline
- New data has to be added and the algorithm trained from scratch.
- Can be automated

# Machine Learning Appoach:Online



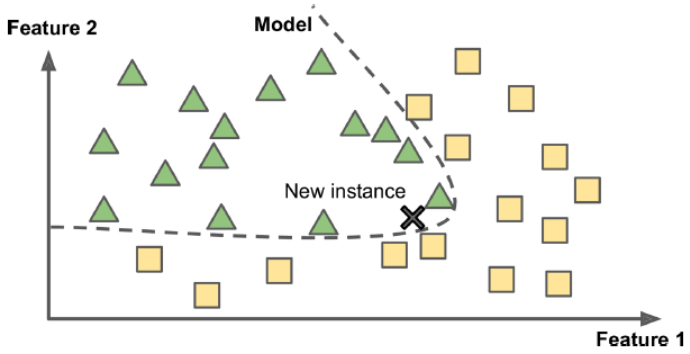

- System can be trained incrementally, either individually or mini batches.
- Can learn about new data on the fly
- Great for continous data.

- One important parameter, how fast it should adapt to new data.
- high learning rate, quick adaptation to new data, but also tends to forget old data.
- Also bad data and, high learning rate, can adversely effect the performance of the system.

# Machine Learning Appoach:Instance based learning


- spam filter could flag similar mails to known spam mails.
- simple similarity measure could be count of common words.
- This is called instance based learning.

# Machine Learning Appoach:Model based learning




- generalize model from a set of samples.
- Use the model to make predictions
- Called model based learning.
- Also called function approximation

# Machine Learning Appoach:Model based learning:loading data

```
fl1 = fileloader("lifesat/oecd_bli_2015.csv")
fl1.fetch_data()
oecd_bli=fl1.load_data()
oecd_bli = oecd_bli[oecd_bli["INEQUALITY"]=="TOT"]
oecd_bli = oecd_bli.pivot(index="Country", columns="Indicator", values="Value")
print("lifesat:\n",oecd_bli.head(2))
print(oecd_bli["Life satisfaction"].head())

fl2 = fileloader("lifesat/gdp_per_capita.csv")
fl2.fetch_data()
gdp_per_capita=fl2.load_data_le()
gdp_per_capita.rename(columns={"2015": "GDP per capita"}, inplace=True)
gdp_per_capita.set_index("Country", inplace=True)
print("GDP/capita:\n",gdp_per_capita.head(2))
full_country_stats = pd.merge(left=oecd_bli, right=gdp_per_capita, left_index=True, right_index=True)
full_country_stats.sort_values(by="GDP per capita", inplace="True")
print(full_country_stats)
print("Canada",full_country_stats[["GDP per capita", 'Life satisfaction']].loc["Canada"])
```

```
Indicator  Air pollution  Assault rate  Consultation on rule-making  \
Country
Australia           13.0           2.1                         10.5
Austria             27.0           3.4                          7.1

Indicator  Dwellings without basic facilities  Educational attainment  \
Country
Australia                                 1.1                    76.0
Austria                                   1.0                    83.0

Indicator  Employees working very long hours  Employment rate  Homicide rate  \
Country
Australia                              14.02            72.0            0.8
Austria                                 7.61            72.0            0.4

Indicator  Household net adjusted disposable income  \
Country
Australia                                  31588.0
Austria                                    31173.0

Indicator  Household net financial wealth      ...      \
Country                                       ...
Australia                        47657.0      ...
Austria                          49887.0      ...
```

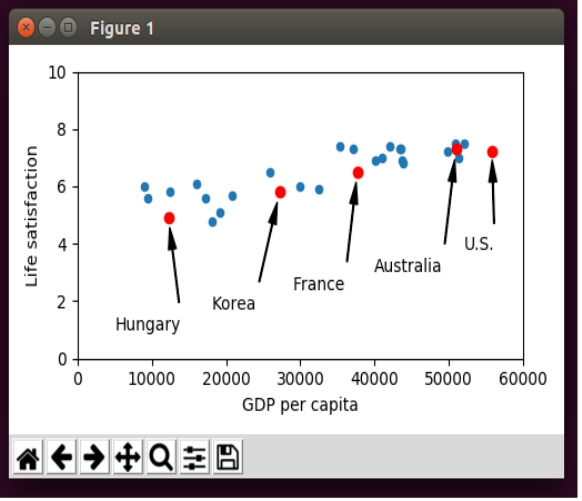
Out[5]:

| Country | Indicator | Air pollution | Assault rate | Consultation on rule-making | Dwellings without basic facilities | Educational attainment | Employees working very long hours | Employment rate | Homicide rate | Household net adjusted disposable income | Household net financial wealth | ... | Time devoted to leisure and personal care | Vote turnou |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Brazil | | 18 | 7.9 | 4.0 | 6.7 | 45 | 10.41 | 67 | 25.5 | 11664 | 6844 | ... | 14.97 | 7 |
| Mexico | | 30 | 12.8 | 9.0 | 4.2 | 37 | 28.83 | 61 | 23.4 | 13085 | 9056 | ... | 13.89 | 6 |

# Machine Learning Appoach:Model based learning

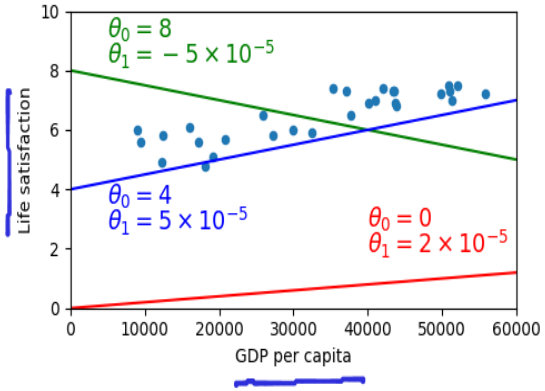
```
fl1.fetch_data()
oecd_bli=fl1.load_data()
oecd_bli = oecd_bli[oecd_bli["INEQUALITY"]=="TOT"]
oecd_bli = oecd_bli.pivot(index="Country", columns="Indicator", values="Value")

fl2 = fileloader("lifesat/gdp_per_capita.csv")
fl2.fetch_data()
gdp_per_capita=fl2.load_data_le()
gdp_per_capita.rename(columns={"2015": "GDP per capita"}, inplace=True)
gdp_per_capita.set_index("Country", inplace=True)

full_country_stats = pd.merge(left=oecd_bli, right=gdp_per_capita, left_index=True, right_index=True)
full_country_stats.sort_values(by="GDP per capita", inplace="True")

remove_indices = [0, 1, 6, 8, 33, 34, 35]
keep_indices = list(set(range(36)) - set(remove_indices))

sample_data = full_country_stats[["GDP per capita", 'Life satisfaction']].iloc[keep_indices]
missing_data = full_country_stats[["GDP per capita", 'Life satisfaction']].iloc[remove_indices]

print("sample data:\n",sample_data,"\n")
print("missing_data:\n",missing_data,"\n")


sample_data.plot(kind='scatter', x="GDP per capita", y='Life satisfaction', figsize=(5,3))
```

```
(ml_home) mohit@nomind:~/Work/ArtificialIntelligence$ ./ML/lifesat/happinessplot.py
sample data:
                  GDP per capita  Life satisfaction
Country
Russia                  9054.914                6.0
Turkey                  9437.372                5.6
Hungary                12239.894                4.9
Poland                 12495.334                5.8
Slovak Republic        15991.736                6.1
Estonia                17288.083                5.6
Greece                 18064.288                4.8
Portugal               19121.592                5.1
Slovenia               20732.482                5.7
Spain                  25864.721                6.5
Korea                  27195.197                5.8
Italy                  29866.581                6.0
Japan                  32485.545                5.9
Israel                 35343.336                7.4
New Zealand            37044.891                7.3
France                 37675.006                6.5
Belgium                40106.632                6.9
Germany                40996.511                7.0
Finland                41973.988                7.4
Canada                 43331.961                7.3
```




- outliers removed for now
- A typical scatter plot.
- Data is noisy but there is a trend.

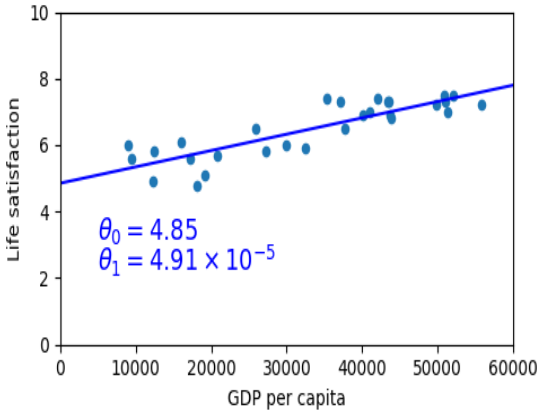
# Machine Learning Appoach:Model based learning




- The model has 2 params $\theta_0, \theta_1$.
- These params have to be "learned" from data using a "fitness" or "cost" function.
- Linear Regression typically uses a cost function.
- This step is called model selection.

# Machine Learning Appoach:Model based learning

```
from sklearn import linear_model
lin1 = linear_model.LinearRegression()
Xsample = np.c_[sample_data["GDP per capita"]]
ysample = np.c_[sample_data["Life satisfaction"]]
lin1.fit(Xsample, ysample)
t0, t1 = lin1.intercept_[0], lin1.coef_[0][0]
print("theta0:",t0, "theta1:",t1)
```

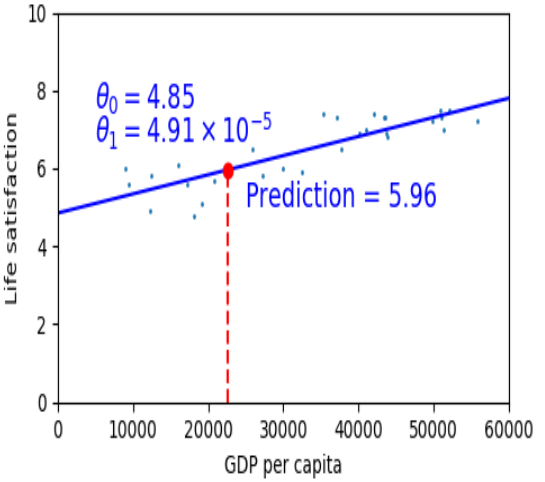



```
(ml_home) mohit@nomind:~/Work/ArtificialIntelligence$ ./ML/lifesat/linearfitplot.py
theta0: 4.85305280027 theta1: 4.91154458916e-05
```

- Now this model can be used to predict life satisfaction for new data.

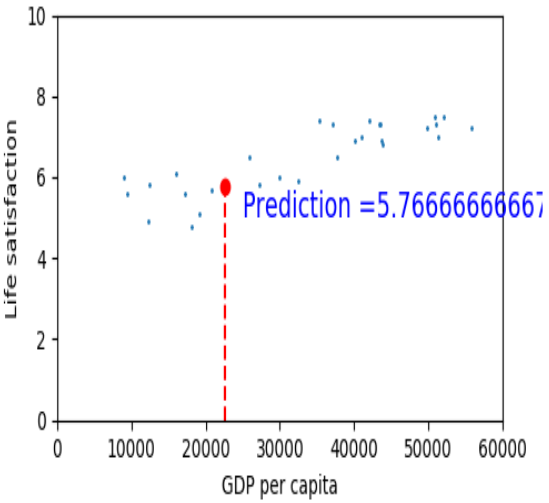
# Machine Learning Appoach:Model based learning:Linear Regression

```
fl2.fetch_data()
gdp_per_capita=fl2.load_data_le()
gdp_per_capita.rename(columns={"2015": "GDP per capita"}, inplace=True)
gdp_per_capita.set_index("Country", inplace=True)

#prepare data
full_country_stats = pd.merge(left=oecd_bli, right=gdp_per_capita, left_index=True, right_index=True)
full_country_stats.sort_values(by="GDP per capita", inplace="True")
remove_indices = [0, 1, 6, 8, 33, 34, 35]
keep_indices = list(set(range(36)) - set(remove_indices))
sample_data = full_country_stats[["GDP per capita", 'Life satisfaction']].iloc[keep_indices]
missing_data = full_country_stats[["GDP per capita", 'Life satisfaction']].iloc[remove_indices]
print("sample:",sample_data,"\n")
print("missing:",missing_data,"\n")
Xsample = np.c_[sample_data["GDP per capita"]]
ysample = np.c_[sample_data["Life satisfaction"]]


#select Model
lin1 = linear_model.LinearRegression()

#train
lin1.fit(Xsample, ysample)
t0, t1 = lin1.intercept_[0], lin1.coef_[0][0]
print("theta0:",t0, "theta1:",t1)

# Make a prediction
cyprus_gdp_per_capita = gdp_per_capita.loc["Cyprus"]["GDP per capita"]
print("cyprus GDP:",cyprus_gdp_per_capita,"\n")
cyprus_predicted_life_satisfaction = lin1.predict(cyprus_gdp_per_capita)[0][0]
print("cyprus:",cyprus_predicted_life_satisfaction,"\n")
```



# Machine Learning Appoach:Model based learning:K-Nearest Neighbours

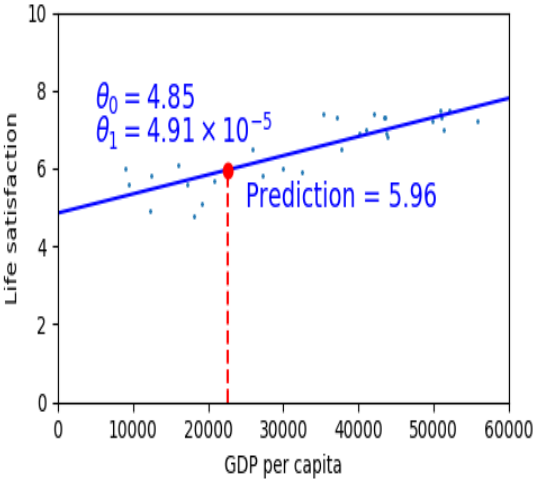
```
missing_data = full_country_stats[["GDP per capita", 'Life satisfaction']].iloc[remove_indices]

print("sample:",sample_data,"\n")
print("missing:",missing_data,"\n")

#select Model
lin1 = neighbors.KNeighborsRegressor(n_neighbors=3)

#prepare data
Xsample = np.c_[sample_data["GDP per capita"]]
ysample = np.c_[sample_data["Life satisfaction"]]

#train
lin1.fit(Xsample, ysample)

cyprus_gdp_per_capita = gdp_per_capita.loc["Cyprus"]["GDP per capita"]
print("cyprus GDP:",cyprus_gdp_per_capita,"\n")

# Make a prediction
cyprus_predicted_life_satisfaction = lin1.predict(cyprus_gdp_per_capita)[0][0]
print("cyprus:",cyprus_predicted_life_satisfaction,"\n")

sample_data.plot(kind='scatter', x="GDP per capita", y='Life satisfaction', figsize=(5,3), s=1)
X=np.linspace(0, 60000, 1000)
plt.axis([0, 60000, 0, 10])
plt.plot([cyprus_gdp_per_capita, cyprus_gdp_per_capita], [0, cyprus_predicted_life_satisfaction], "r--")
plt.text(25000, 5.0, r"Prediction ="+str(cyprus_predicted_life_satisfaction), fontsize=14, color="b")
plt.plot(cyprus_gdp_per_capita, cyprus_predicted_life_satisfaction, "ro")
fl1.save_fig('cyprus_prediction_plot')
plt.show()
```




- different model

# Machine Learning Appoach:Model based learning:Linear Regression

```
fl2.fetch_data()
gdp_per_capita=fl2.load_data_le()
gdp_per_capita.rename(columns={"2015": "GDP per capita"}, inplace=True)
gdp_per_capita.set_index("Country", inplace=True)

#prepare data
full_country_stats = pd.merge(left=oecd_bli, right=gdp_per_capita, left_index=True, right_index=True)
full_country_stats.sort_values(by="GDP per capita", inplace="True")
remove_indices = [0, 1, 6, 8, 33, 34, 35]
keep_indices = list(set(range(36)) - set(remove_indices))
sample_data = full_country_stats[["GDP per capita", 'Life satisfaction']].iloc[keep_indices]
missing_data = full_country_stats[["GDP per capita", 'Life satisfaction']].iloc[remove_indices]
print("sample:",sample_data,"\n")
print("missing:",missing_data,"\n")
Xsample = np.c_[sample_data["GDP per capita"]]
ysample = np.c_[sample_data["Life satisfaction"]]


#select Model
lin1 = linear_model.LinearRegression()

#train
lin1.fit(Xsample, ysample)
t0, t1 = lin1.intercept_[0], lin1.coef_[0][0]
print("theta0:",t0, "theta1:",t1)

# Make a prediction
cyprus_gdp_per_capita = gdp_per_capita.loc["Cyprus"]["GDP per capita"]
print("cyprus GDP:",cyprus_gdp_per_capita,"\n")
cyprus_predicted_life_satisfaction = lin1.predict(cyprus_gdp_per_capita)[0][0]
print("cyprus:",cyprus_predicted_life_satisfaction,"\n")
```

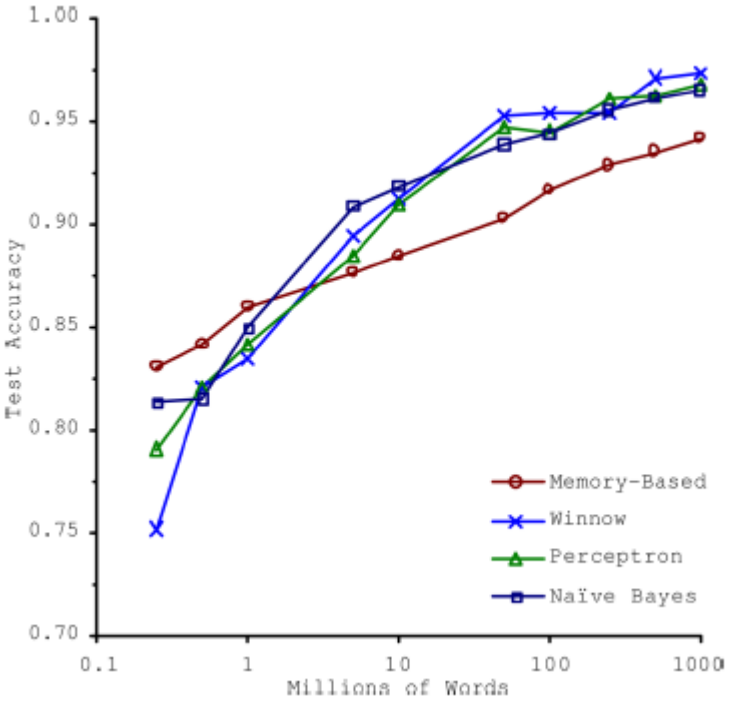


# Machine Learning Appoach:Challenges:

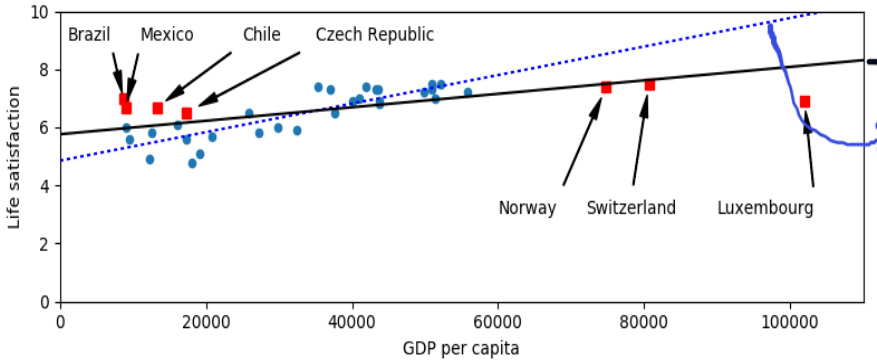
- Insufficient Quantity of Training Data

# Machine Learning Appoach:Challenges:

- The Unreasonable Effectiveness of Data



# Machine Learning Appoach:Challenges:Non-representative data




→ all data

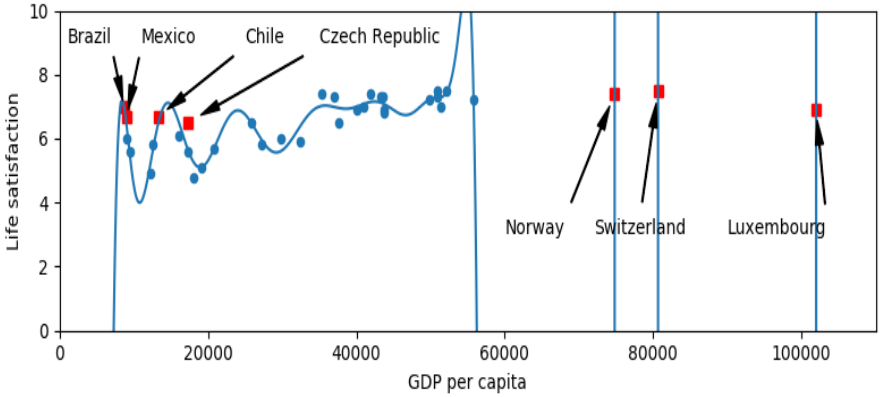
→ without outliers

- A simple linear model may not work well.
- crucial to use sample representative of the cases to generalize for.
- while there may be genuine outliers, often some of them can be accounted for by adding more features or a combination of features.

# Machine Learning Appoach:Challenges

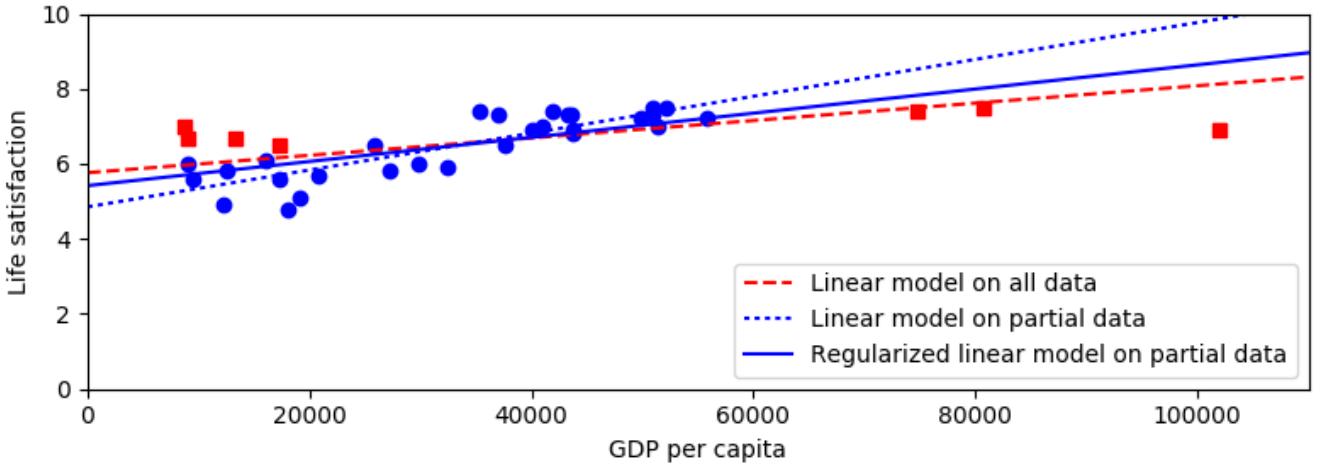
- Sampling Bias
- Poor Data Quality
  - Well worth the effort to clean data
- Irrelevant features

# Machine Learning Appoach:Challenges:Overfitting




- ML can also fall into the trap of overgeneralizing.
- This is called overfitting
- It performs much better on training data than on testing data.

# Machine Learning Appoach:Challenges:Overfitting




- regularization is often a trick to mitigate the effects of overfitting.

# Machine Learning Appoach:Challenges:Underfitting

- The main options to fix this problem are:
  - Selecting a more powerful model, with more parameters
  - Feeding better features to the learning algorithm (feature engineering)
  - Reducing the constraints on the model (e.g., reducing the regularization hyperparameter)

# Machine Learning Appoach:Challenges

- The only way to know how well a model will generalize to new cases is to actually try it out on new cases.
- A better option is to split your data into two sets: the training set and the test set. As these names imply, you train your model using the training set, and you test it using the test set.