Arun Manglick - Artificial Intelligence & Machine/Deep Learn

Thanks for Visit!!!

My Blogs...

Arun Manglick - Technical View
Async Await...C# 5.0

Arun Manglick - Blockchain & Bitcoin
Cardano - New Smart Contract Platform

Arun Manglick - Project Management
Scrum/Kanban - Adding User Story Between Sprint

Arun Manglick - Architect View
Collection of Visio Shapes:

Arun Manglick - Artificial Intelligence & Machine/Deep Learning
Microsoft Cognitive - Overview

Arun Manglick - Cloud Computing
10 New AWS Cloud Services - Till Jun 2017

Arun Manglick - About Me
Practices - Summary

About Me

Arun Manglick

Here is my profile – AWS-ASA/Dev/SysOps, SAFe Agilist, PMP, PMI-ACP, CSM, PRINCE2 Practitioner, MS-Project, CSSGB, ITIL, MCTS,MCPD, M.Tech, MCA. Well with 15+ years of Experience, I'm in the role of Project Manager (Technical Manager) with an MNC. My expertise lies in AWS Cloud & Microsoft platform. Worked in ASP.Net, ASP.NET MVC 3.0, SharePoint 2010, Silverlight 4.0, PRISM, WCF 4.0 , REST, RIA Services with C#. Explored the latest technologies JQuery, LINQ 2 SQL, LINQ 2 Entity - ADO.Net Entity Framework, etc.. Along with technology, I also focus on Software Architecture & Design. Besides being Technical, Project Management is my focus area. Managed and Managing many small to medium sized projects. Worked for almost 7+ years in Agile methodology (Extreme Programming & SCRUM).

View my complete profile

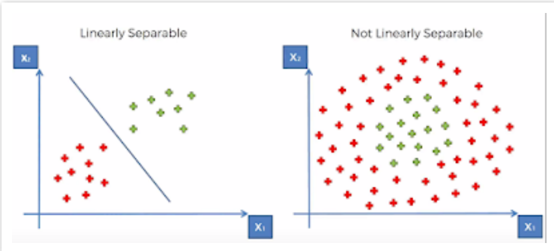
▼ 2017 (41)
► August 2017 (2)
▼ July 2017 (29)
XGBoost
Model Selection
Part 10: Model Selection & Boosting
Kernel PCA
Linear Discriminant Analysis (LDA)

Wednesday, July 5, 2017

Kernel SVM (Support Vector Machine)

Basics:

We see in SVM, data is linearly separable. However, what if the data is not linearly separable, as below. In such cases SVM alone to use Kernel SVM.



Here we'll see how to Map Non Linearly Separable data and map it to Higher Dimensional Space and get a Linearly Separable da Build a Decision Boundary and Project it back to Linear Separable Model. This is done using 'Gaussian RBF Kernel' method.

In machine learning, the Gaussian RBF kernel method, is a popular kernel function used in various kernelized learning algorithms. used in Support Vector Machine classification.

The RBF kernel on two samples x and x', represented as feature vectors in some input space, is defined as below.

Here L is the Landmark.
This function determines
 - Any point close to zero will be in blue area.
 - Any point close to one will be in colored area.

This way we can separate the two non-linear areas.
Value of Sigma determines the circumference of the cone.
 - More the value of Sigma more values will be close to one (Fig 3)
 - Less the value of Sigma more values will be close to zero (Fig 4)

The Kernel method can be used for multiple areas also (Fig 5).

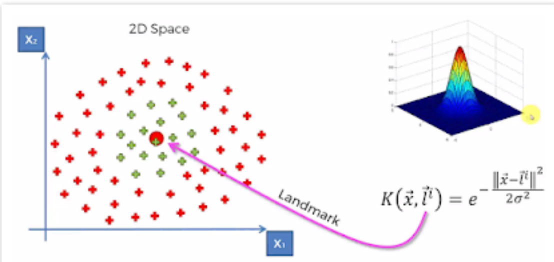


Fig1

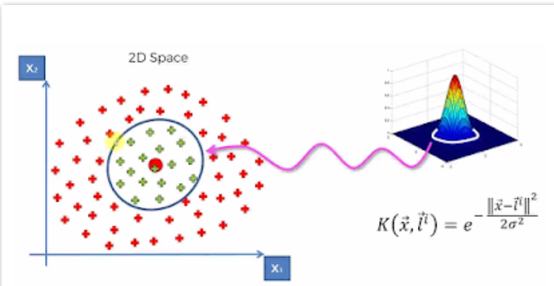


Fig2

- Principal Component Analysis (PCA)
- Part 9: Dimensionality Reduction
- Convolutional Neural Networks
- Artificial Neural Networks
- Part 8: Deep Learning
- Part 7: Natural Language Processing
- Thompson Sampling
- Upper Confidence Bound (UCB)
- Part 6: Reinforcement Learning
- Eclat
- Apriori
- Part 5 - Association Rule Learning
- Hierarchical Clustering
- K-Means Clustering
- Part 4 - Clustering
- Evaluating Classification Models Performance
- Random Forest : Classification
- Decision Tree : Classification
- Naive Bayes' Theorem
- Kernel SVM (Support Vector Machine)
- Support Vector Machine (SVM)
- K-Nearest Neighbors (K-NN)
- Logistic Regression
- Part 3 - Classification

► June 2017 (10)

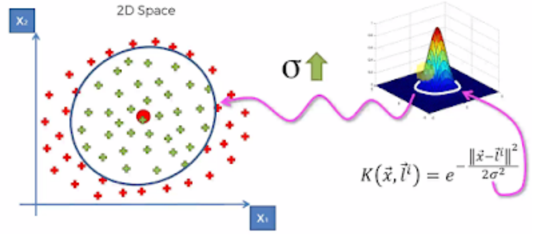


Fig3

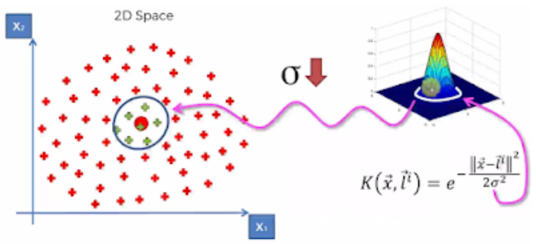


Fig4

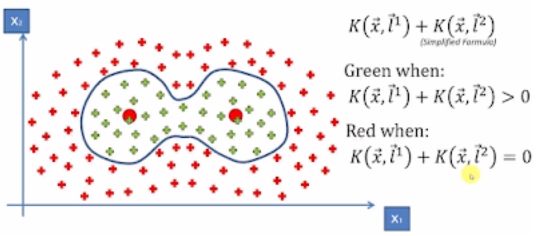
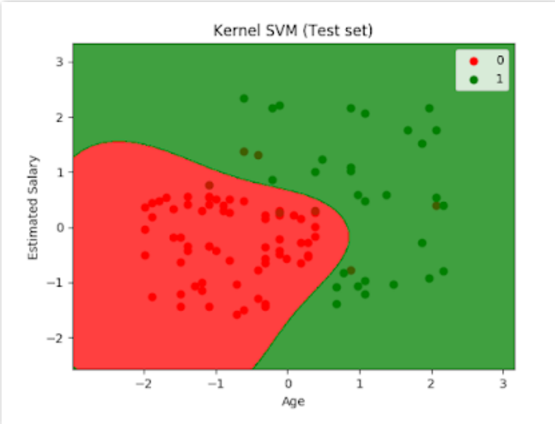


Fig5

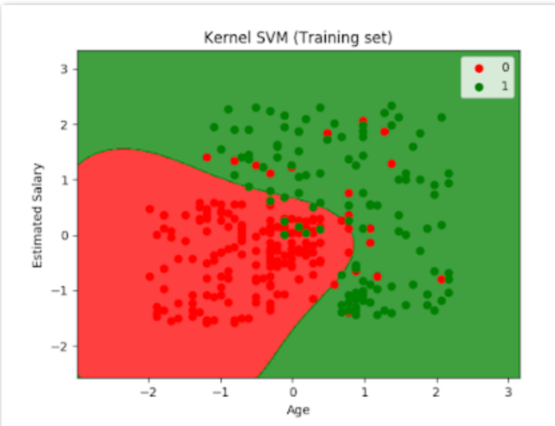
Code: Kernel SVM

```
# Importing the libraries
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd

# Importing the dataset
dataset = pd.read_csv('Social_Network_Ads.csv')
X = dataset.iloc[:, [2, 3]].values
y = dataset.iloc[:, 4].values

# Splitting the dataset into the Training set and Test set
from sklearn.cross_validation import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.25, random_state = 0)

# Feature Scaling
from sklearn.preprocessing import StandardScaler
sc = StandardScaler()
X_train = sc.fit_transform(X_train)
X_test = sc.transform(X_test)

# Fitting Kernel SVM to the Training set
from sklearn.svm import SVC
classifier = SVC(kernel = 'rbf', random_state = 0)
classifier.fit(X_train, y_train)

# Predicting the Test set results
y_pred = classifier.predict(X_test)

# Making the Confusion Matrix
# Used to evauate performance of model to see corret/incorrection predictions made by Logistic regression
from sklearn.metrics import confusion_matrix
cm = confusion_matrix(y_test, y_pred)

# Visualising the Training set results
from matplotlib.colors import ListedColormap
X_set, y_set = X_train, y_train
X1, X2 = np.meshgrid(np.arange(start = X_set[:, 0].min() - 1, stop = X_set[:, 0].max() + 1, step = 0.01),
                     np.arange(start = X_set[:, 1].min() - 1, stop = X_set[:, 1].max() + 1, step = 0.01))
```

```
plt.contourf(X1, X2, classifier.predict(np.array([X1.ravel(), X2.ravel()]).T).reshape(X1.shape),
             alpha = 0.75, cmap = ListedColormap(('red', 'green')))
plt.xlim(X1.min(), X1.max())
plt.ylim(X2.min(), X2.max())
for i, j in enumerate(np.unique(y_set)):
    plt.scatter(X_set[y_set == j, 0], X_set[y_set == j, 1],
                c = ListedColormap(('red', 'green'))(i), label = j)
plt.title('Kernel SVM (Training set)')
plt.xlabel('Age')
plt.ylabel('Estimated Salary')
plt.legend()
plt.show()
```




```
# Visualising the Test set results
from matplotlib.colors import ListedColormap
X_set, y_set = X_test, y_test
X1, X2 = np.meshgrid(np.arange(start = X_set[:, 0].min() - 1, stop = X_set[:, 0].max() + 1, step = 0.01),
                     np.arange(start = X_set[:, 1].min() - 1, stop = X_set[:, 1].max() + 1, step = 0.01))
plt.contourf(X1, X2, classifier.predict(np.array([X1.ravel(), X2.ravel()]).T).reshape(X1.shape),
             alpha = 0.75, cmap = ListedColormap(('red', 'green')))
plt.xlim(X1.min(), X1.max())
plt.ylim(X2.min(), X2.max())
for i, j in enumerate(np.unique(y_set)):
    plt.scatter(X_set[y_set == j, 0], X_set[y_set == j, 1],
                c = ListedColormap(('red', 'green'))(i), label = j)
plt.title('Kernel SVM (Test set)')
plt.xlabel('Age')
plt.ylabel('Estimated Salary')
plt.legend()
plt.show()
```



Hope this helps !

Arun Manglick

---

at July 05, 2017 G+

No comments:

Post a Comment

Enter your comment...

Comment as: Google Accoun

Publish Preview

Newer Post Home

Subscribe to: Post Comments (Atom)