

## Contents

<b>1</b>	<b>Top Index</b>	<b>1</b>
<b>2</b>	<b>Anoop index:</b>	<b>2</b>
<b>3</b>	<b>(edited to top anoop, rest of page) Applied Machine Learning Process</b>	<b>3</b>
<b>4</b>	<b>experimental0</b>	<b>19</b>
<b>5</b>	<b>How Do I Get Started?</b>	<b>19</b>
5.1	articles-read, worth rereading . . . . .	19
5.2	notes . . . . .	20
5.2.1	intro . . . . .	20
5.2.2	some key word definitions: . . . . .	20
<b>6</b>	<b><a href="https://machinelearningmastery.com/python-machine-learning-mini-course/">https://machinelearningmastery.com/python-machine-learning-mini-course/</a></b>	<b>22</b>
6.1	pre requisite <a href="https://machinelearningmastery.com/gentle-introduction-to-the-bias-v">https://machinelearningmastery.com/gentle-introduction-to-the-bias-v</a>	
6.2	1 - install . . . . .	22
6.2.1	next time try using this tutorial <a href="https://sourabhbajaj.com/mac-setup/Python/numpy.html">https://sourabhbajaj.com/mac-setup/Python/numpy.html</a> . . . . .	22
6.2.2	make a new virtualenv . . . . .	22
6.2.3	install <a href="https://stackoverflow.com/questions/26319762/how-to-install-scipy-stack-with-pip-and-homebrew">https://stackoverflow.com/questions/26319762/how-to-install-scipy-stack-with-pip-and-homebrew</a> . . . . .	23
6.2.4	check if properly isntalled using <code>__version__</code> after import . . . . .	23
6.3	2 - python, pandas, numpy, matplotlib veeery basics . . . . .	23
6.3.1	python, also refer in-y-minutes file for future reference . . . . .	23
6.3.2	numpy basics . . . . .	25
6.3.3	matplotlib basics . . . . .	29
6.3.4	Pandas -basics: . . . . .	30
6.4	3 - Load csv . . . . .	31
6.5	4 - use pandas.DataFrame helper functions to describe data with statistics . . . . .	32
6.6	5 - Basic Data Visualization . . . . .	33

## 1 Top Index

Foundations

	<i>HowDoIGetStarted?</i>
Step-by-Step Process	
	<i>LinearAlgebra</i>
Statistical Methods	
Beginner	
	<i>UnderstandAlgorithms</i>
Weka ML (no code)	
	<i>PythonML(scikit – learn)</i>
R ML (caret)	
Intermediate	
	<i>CodeAlgorithms(Python)</i>
Time Series Forecasting	
	<i>XGBoostAlgorithm</i>
Deep Learning (Keras)	
Advanced	
	<i>DeepLearningandLSTMs</i>
Deep Learning for Text	Deep Learning for Time Series Forecasting

## 2 Anoop index:

How Do I Get Started? Applied Machine Learning Process Linear Algebra Statistical Methods Understand Machine Learning Algorithms Python Machine Learning (scikit-learn) Code Algorithm from Scratch (Python) Introduction to Time Series Forecasting (Python) XGBoost in Python (Stochastic Gradient Boosting) Deep Learning (Keras) Long Short-Term Memory (LSTM) Deep Learning for Natural Language Processing (NLP) Deep Learning for Time Series Forecasting

### 3 (edited to top anoop, rest of page) Applied Machine Learning Process

The benefit of machine learning are the predictions and the models that make predictions.

To have skill at applied machine learning means knowing how to consistently and reliably deliver high-quality predictions on problem after problem. You need to follow a systematic process.

Below is a 5-step process that you can follow to consistently achieve above average results on predictive modeling problems:

\$\$ Step 1: Define your problem.

\$\$ How to Define Your Machine Learning Problem

\$\$ Step 2: Prepare your data.

*HowtoPrepareDataForMachineLearning*

How to Identify Outliers in your Data

*ImproveModelAccuracywithDataPre – Processing*

Discover Feature Engineering

*AnIntroductiontoFeatureSelection*

Tactics to Combat Imbalanced Classes in Your Machine Learning Dataset

\$\$ Data Leakage in Machine Learning

\$\$ Step 3: Spot-check algorithms.

*HowtoEvaluateMachineLearningAlgorithms*

Why you should be Spot-Checking Algorithms on your Machine Learning Problems

*HowToChooseTheRightTestOptionsWhenEvaluatingMachineLearningAlgorithms*

A Data-Driven Approach to Choosing Machine Learning Algorithms

\$\$ Step 4: Improve results.

*HowtoImproveMachineLearningResults*

Machine Learning Performance Improvement Cheat Sheet \$\$ How To Improve Deep Learning Performance

\$\$ Step 5: Present results.

### *HowtoUseMachineLearningResults*

How to Train a Final Machine Learning Model \$\$ How To Deploy Your Predictive Model To Production

For a good summary of this process, see the posts:

### *AppliedMachineLearningProcess*

How to Use a Machine Learning Checklist to Get Accurate Predictions

Linear Algebra

Linear algebra is an important foundation area of mathematics required for achieving a deeper understanding of machine learning algorithms.

Below is the 3 step process that you can use to get up-to-speed with linear algebra for machine learning, fast.

\$\$ Step 1: Discover what Linear Algebra is.

\$\$ A Gentle Introduction to Linear Algebra

\$\$ Step 2: Discover why Linear Algebra is important for machine learning.

### *5ReasonstoLearnLinearAlgebraforMachineLearning*

Linear Algebra for Machine Learning

\$\$ Step 3: Dive into Linear Algebra topics.

### *LinearAlgebraforMachineLearningMini – Course*

Linear Algebra for Machine Learning (my book)

You can see all linear algebra posts here. Below is a selection of some of the most popular tutorials.

Linear Algebra in Python

### *IntroductiontoN – DimensionalArraysinPython*

How to Index, Slice and Reshape NumPy Arrays

Matrices

### *IntroductiontoMatricesandMatrixArithmetic*

Introduction to Matrix Types in Linear Algebra

Vectors

### *IntroductiontoVectors*

Introduction to Vector Norms  
Matrix Factorization

*Introduction to Matrix Factorization*

Introduction to Eigendecomposition

Statistical Methods

Statistical Methods an important foundation area of mathematics required for achieving a deeper understanding of the behavior of machine learning algorithms.

Below is the 3 step process that you can use to get up-to-speed with statistical methods for machine learning, fast.

\$\$ Step 1: Discover what Statistical Methods are.

\$\$ What is Statistics (and why is it important in machine learning)?

\$\$ Step 2: Discover why Statistical Methods are important for machine learning.

*The Close Relationship Between Applied Statistics and Machine Learning*

10 Examples of How to Use Statistical Methods in a Machine Learning Project

\$\$ Step 3: Dive into the topics of Statistical Methods.

*Statistics for Machine Learning (7 – Day Mini – Course)*

Statistical Methods for Machine Learning (my book)

You can see all of the statistical methods posts here. Below is a selection of some of the most popular tutorials.

Summary Statistics

*Introduction to the 5 Number Summary*

Introduction to Data Visualization

Hypothesis Tests

*15 Hypothesis Tests (cheatsheet)*

Tests for Comparing Algorithms

Resampling Methods

*Introduction to the Bootstrap*

Introduction to Cross-Validation  
Estimation Statistics

*Introduction to Estimation Statistics*

Introduction to Confidence Intervals

Understand Machine Learning Algorithms

Machine learning is about machine learning algorithms.

You need to know what algorithms are available for a given problem, how they work, and how to get the most out of them.

Here's how to get started with machine learning algorithms:

\$\$ Step 1: Discover the different types of machine learning algorithms.

\$\$ A Tour of Machine Learning Algorithms

\$\$ Step 2: Discover the foundations of machine learning algorithms.

*How Machine Learning Algorithms Work*

Parametric and Nonparametric Algorithms

*Supervised and Unsupervised Algorithms*

The Bias-Variance Trade-Off \$\$ Overfitting and Underfitting With Algorithms

\$\$ Step 3: Discover how top machine learning algorithms work.

*Machine Learning Algorithms Mini – Course*

Master Machine Learning Algorithms (my book)

You can see all machine learning algorithm posts here. Below is a selection of some of the most popular tutorials.

Linear Algorithms

*Gradient Descent*

Linear Regression

*Logistic Regression*

Linear Discriminant Analysis

Nonlinear Algorithms

*Classification and Regression Trees*

Naive Bayes

*K – Nearest Neighbors*

Learning Vector Quantization \$\$ Support Vector Machines  
Ensemble Algorithms

*BaggingandRandomForest*

Boosting and AdaBoost

Weka Machine Learning (no code)

Weka is a platform that you can use to get started in applied machine learning.

It has a graphical user interface meaning that no programming is required and it offers a suite of state of the art algorithms.

Heres how you can get started with Weka:

\$\$ Step 1: Discover the features of the Weka platform.

\$\$ What is the Weka Machine Learning Workbench

\$\$ Step 2: Discover how to get around the Weka platform.

*HowtoDownloadandInstalltheWekaMachineLearningWorkbench*

A Tour of the Weka Machine Learning Workbench

\$\$ Step 3: Discover how to deliver results with Weka.

*HowtoRunYourFirstClassifierinWeka*

Applied Machine Learning With Weka Mini-Course \$\$ Machine Learning  
Mastery With Weka (my book)

You can see all Weka machine learning posts here. Below is a selection of some of the most popular tutorials.

Prepare Data in Weka

*HowToLoadCSVMachineLearningDatainWeka*

How to Better Understand Your Machine Learning Data in Weka

*HowtoNormalizeandStandardizeYourMachineLearningDatainWeka*

How To Handle Missing Values In Machine Learning Data With Weka \$\$

How to Perform Feature Selection With Machine Learning Data in Weka

Weka Algorithm Tutorials

*HowtoUseMachineLearningAlgorithmsinWeka*

How To Estimate The Performance of Machine Learning Algorithms in Weka

*HowToUseRegressionMachineLearningAlgorithmsinWeka*

How To Use Classification Machine Learning Algorithms in Weka \$\$ How to Tune Machine Learning Algorithms in Weka

Python Machine Learning (scikit-learn)

Python is one of the fastest growing platforms for applied machine learning.

You can use the same tools like pandas and scikit-learn in the development and operational deployment of your model.

Below are the steps that you can use to get started with Python machine learning:

\$\$ Step 1: Discover Python for machine learning

\$\$ A Gentle Introduction to Scikit-Learn: A Python Machine Learning Library

\$\$ Step 2: Discover the ecosystem for Python machine learning.

*CrashCourseinPythonforMachineLearningDevelopers*

Python Ecosystem for Machine Learning \$\$ Python is the Growing Platform for Applied Machine Learning

\$\$ Step 3: Discover how to work through problems using machine learning in Python.

*YourFirstMachineLearningProjectinPythonStep – By – Step*

Python Machine Learning Mini-Course \$\$ Machine Learning Mastery With Python (my book)

You can see all Python machine learning posts here. Below is a selection of some of the most popular tutorials.

Prepare Data in Python

*HowToLoadMachineLearningDatainPython*

Understand Your Machine Learning Data With Descriptive Statistics in Python

*VisualizeMachineLearningDatainPythonWithPandas*

How To Prepare Your Data For Machine Learning in Python with Scikit-Learn \$\$ Feature Selection For Machine Learning in Python



Machine Learning in Python

*Evaluate the Performance of Machine Learning Algorithms*

Metrics To Evaluate Machine Learning Algorithms in Python

*Spot-Check Classification Machine Learning Algorithms in Python with scikit-learn*

Spot-Check Regression Machine Learning Algorithms in Python with scikit-learn  
\$\$ How To Compare Machine Learning Algorithms in Python with scikit-learn

R Machine Learning (caret)

R is a platform for statistical computing and is the most popular platform among professional data scientists.

Its popular because of the large number of techniques available, and because of excellent interfaces to these methods such as the powerful caret package.

Heres how to get started with R machine learning:

\$\$ Step 1: Discover the R platform and why it is so popular.

*What is R*

Use R For Machine Learning \$\$ Super Fast Crash Course in R

\$\$ Step 2: Discover machine learning algorithms in R.

\$\$ How To Get Started With Machine Learning Algorithms in R

\$\$ Step 3: Discover how to work through problems using machine learning in R.

*Your First Machine Learning Project in R Step – By – Step*

R Machine Learning Mini-Course \$\$ Machine Learning Mastery With R (my book)

You can see all R machine learning posts here. Below is a selection of some of the most popular tutorials.

Data Preparation in R

*How To Load Your Machine Learning Data Into R*

Better Understand Your Data in R Using Descriptive Statistics

*Better Understand Your Data in R Using Visualization*

Feature Selection with the Caret R Package \$\$ Get Your Data Ready For  
Machine Learning in R with Pre-Processing  
Applied Machine Learning in R

### *HowtoEvaluateMachineLearningAlgorithmswithR*

Spot Check Machine Learning Algorithms in R

### *TuneMachineLearningAlgorithmsinR*

How to Build an Ensemble Of Machine Learning Algorithms in R \$\$ Com-  
pare The Performance of Machine Learning Algorithms in R

Code Algorithm from Scratch (Python)

You can learn a lot about machine learning algorithms by coding them  
from scratch.

Learning via coding is the preferred learning style for many developers  
and engineers.

Heres how to get started with machine learning by coding everything  
from scratch.

\$\$ Step 1: Discover the benefits of coding algorithms from scratch.

### *BenefitsofImplementingMachineLearningAlgorithmsFromScratch*

Understand Machine Learning Algorithms By Implementing Them From  
Scratch

\$\$ Step 2: Discover that coding algorithms from scratch is a learning  
tool only.

\$\$ Stop Coding Machine Learning Algorithms From Scratch

\$\$ Step 3: Discover how to code machine learning algorithms from scratch  
in Python.

\$\$ Machine Learning Algorithms From Scratch (my book)

You can see all of the Code Algorithms from Scratch posts here. Below  
is a selection of some of the most popular tutorials.

Prepare Data

### *HowtoLoadMachineLearningDataFromScratch*

How to Scale Machine Learning Data From Scratch

Linear Algorithms

### *HowToImplementSimpleLinearRegressionFromScratch*

How To Implement The Perceptron Algorithm From Scratch  
Algorithm Evaluation

*HowtoCodeResamplingMethodsFromScratch*

How To Code Algorithm Performance Metrics From Scratch  
Nonlinear Algorithms

*HowtoCodeTheBackpropagationAlgorithmFromScratch*

How To Code The Decision Tree Algorithm From Scratch

Introduction to Time Series Forecasting (Python)

Time series forecasting is an important topic in business applications.

Many datasets contain a time component, but the topic of time series is rarely covered in much depth from a machine learning perspective.

Heres how to get started with Time Series Forecasting:

\$\$ Step 1: Discover Time Series Forecasting.

\$\$ What Is Time Series Forecasting?

\$\$ Step 2: Discover Time Series as Supervised Learning.

\$\$ Time Series Forecasting as Supervised Learning

\$\$ Step 3: Discover how to get good at delivering results with Time Series Forecasting.

*TimeSeriesForecastingWithPythonMini – Course*

Time Series Forecasting With Python (my book)

You can see all Time Series Forecasting posts here. Below is a selection of some of the most popular tutorials.

Data Preparation Tutorials

*7TimeSeriesDatasetsforMachineLearning*

How to Load and Explore Time Series Data in Python

*HowtoNormalizeandStandardizeTimeSeriesDatainPython*

Basic Feature Engineering With Time Series Data in Python \$\$ How To  
Backtest Machine Learning Models for Time Series Forecasting  
Forecasting Tutorials

*HowtoMakeBaselinePredictionsforTimeSeriesForecastingwithPython*

How to Check if Time Series Data is Stationary with Python

*How to Create an ARIMA Model for Time Series Forecasting with Python*

How to Grid Search ARIMA Model Hyperparameters with Python \$\$ How to Work Through a Time Series Forecast Project

XGBoost in Python (Stochastic Gradient Boosting)

XGBoost is a highly optimized implementation of gradient boosted decision trees.

It is popular because it is being used by some of the best data scientists in the world to win machine learning competitions.

Here's how to get started with XGBoost:

\$\$ Step 1: Discover the Gradient Boosting Algorithm.

\$\$ A Gentle Introduction to the Gradient Boosting Algorithm for Machine Learning

\$\$ Step 2: Discover XGBoost.

\$\$ A Gentle Introduction to XGBoost for Applied Machine Learning

\$\$ Step 3: Discover how to get good at delivering results with XGBoost.

*How to Develop Your First XGBoost Model in Python with scikit – learn*

XGBoost With Python Mini-Course \$\$ XGBoost With Python (my book)

You can see all XGBoosts posts here. Below is a selection of some of the most popular tutorials.

XGBoost Basics

*Data Preparation for Gradient Boosting with XGBoost in Python*

How to Evaluate Gradient Boosting Models with XGBoost in Python

*Avoid Overfitting By Early Stopping With XGBoost In Python*

Feature Importance and Feature Selection With XGBoost in Python

XGBoost Tuning

*How to Configure the Gradient Boosting Algorithm*

Tune Learning Rate for Gradient Boosting with XGBoost in Python

*Stochastic Gradient Boosting with XGBoost and scikit – learn in Python*

How to Tune the Number and Size of Decision Trees with XGBoost in Python

\$\$ How to Best Tune Multithreading Support for XGBoost in Python

Deep Learning (Keras)

Deep learning is a fascinating and powerful field.

State-of-the-art results are coming from the field of deep learning and it is a sub-field of machine learning that cannot be ignored.

Heres how to get started with deep learning:

\$\$ Step 1: Discover what deep learning is all about.

*What is Deep Learning?*

8 Inspirational Applications of Deep Learning

\$\$ Step 2: Discover the best tools and libraries.

*Introduction to the Python Deep Learning Library Theano*

Introduction to the Python Deep Learning Library TensorFlow \$\$ Introduction to Python Deep Learning with Keras

\$\$ Step 3: Discover how to work through problems and deliver results.

*Develop Your First Neural Network in Python With Keras Step – By – Step*

Applied Deep Learning in Python Mini-Course \$\$ Deep Learning With Python (my book)

You can see all deep learning posts here. Below is a selection of some of the most popular tutorials.

Background

*Crash Course On Multi – Layer Perceptron Neural Networks*

Crash Course in Convolutional Neural Networks for Machine Learning \$\$

Crash Course in Recurrent Neural Networks for Deep Learning

Multilayer Perceptrons

*5 Step Life – Cycle for Neural Network Models in Keras*

How to Grid Search Hyperparameters for Deep Learning Models in Python With Keras

*Save and Load Your Keras Deep Learning Models*

Display Deep Learning Model Training History in Keras \$\$ Dropout Regularization in Deep Learning Models With Keras  
Convolutional Neural Networks

*Handwritten Digit Recognition using Convolutional Neural Networks in Python with Keras*

Object Recognition with Convolutional Neural Networks in the Keras Deep Learning Library \$\$ Predict Sentiment From Movie Reviews Using Deep Learning  
Recurrent Neural Networks

*Time Series Prediction with LSTM Recurrent Neural Networks in Python with Keras*

Understanding Stateful LSTM Recurrent Neural Networks in Python with Keras \$\$ Text Generation With LSTM Recurrent Neural Networks in Python with Keras

Long Short-Term Memory (LSTM)

Long Short-Term Memory (LSTM) Recurrent Neural Networks are designed for sequence prediction problems and are a state-of-the-art deep learning technique for challenging prediction problems.

Here's how to get started with LSTMs in Python:

\$\$ Step 1: Discover the promise of LSTMs.

\$\$ The Promise of Recurrent Neural Networks for Time Series Forecasting

\$\$ Step 2: Discover where LSTMs are useful.

*Making Predictions with Sequences*

A Gentle Introduction to Long Short-Term Memory Networks by the Experts

\$\$ Introduction to Models for Sequence Prediction

\$\$ Step 3: Discover how to use LSTMs on your project.

*The 5 Step Life – Cycle for Long Short – Term Memory Models in Keras*

Long Short-Term Memory Networks (Mini-Course) \$\$ Long Short-Term Memory Networks With Python (my book)

You can see all LSTM posts here. Below is a selection of some of the most popular tutorials using LSTMs in Python with the Keras deep learning library.

Data Preparation for LSTMs

*How to Reshape Input Data for Long Short – Term Memory Networks*

How to One Hot Encode Sequence Data

*How to Remove Trends and Seasonality with a Difference Transform*

How to Scale Data for Long Short-Term Memory Networks

*How to Prepare Sequence Prediction for Truncated BPTT*

How to Handle Missing Timesteps in Sequence Prediction Problems  
LSTM Behaviour

*A Gentle Introduction to Backpropagation Through Time*

Demonstration of Memory with a Long Short-Term Memory Network

*How to Use the Time Distributed Layer for Long Short – Term Memory Networks*

How to use an Encoder-Decoder LSTM to Echo Sequences of Random Integers  
Attention in Long Short-Term Memory Recurrent Neural Networks  
Modeling with LSTMs

*Generative Long Short – Term Memory Networks*

Stacked Long Short-Term Memory Networks

*Encoder – Decoder Long Short – Term Memory Networks*

CNN Long Short-Term Memory Networks

*Diagnose Overfitting and Underfitting of LSTM Models*

How to Make Predictions with Long Short-Term Memory Models  
LSTM for Time Series

*On the Suitability of LSTMs for Time Series Forecasting*

Time Series Forecasting with the Long Short-Term Memory Network

*Multi – step Time Series Forecasting with Long Short – Term Memory Networks*

Multivariate Time Series Forecasting with LSTMs in Keras

Deep Learning for Natural Language Processing (NLP)

Working with text data is hard because of the messy nature of natural language.

Text is not solved but to get state-of-the-art results on challenging NLP problems, you need to adopt deep learning methods

Heres how to get started with deep learning for natural language processing:

\$\$ Step 1: Discover what deep learning for NLP is all about.

*What is Natural Language Processing?*

What is Deep Learning? \$\$ Promise of Deep Learning for Natural Language Processing

\$\$ Step 2: Discover standard datasets for NLP.

*7 Applications of Deep Learning for Natural Language Processing*

Datasets for Natural Language Processing

\$\$ Step 3: Discover how to work through problems and deliver results.

*Crash – Course in Deep Learning for Natural Language Processing*

Deep Learning for Natural Language Processing (my book)

You can see all deep learning for NLP posts here. Below is a selection of some of the most popular tutorials.

Bag-of-Words Model

*What is the Bag – of – Words Model?*

How to Prepare Text Data for Machine Learning with scikit-learn \$\$ How to Develop a Bag-of-Words Model for Predicting Sentiment

Language Modeling

*Gentle Introduction to Statistical Language Modeling and Neural Language Models*

How to Develop a Character-Based Neural Language Model in Keras \$\$ How to Develop a Word-Level Neural Language Model and Use it to Generate Text



Text Summarization

*AGentleIntroductiontoTextSummarization*

How to Prepare News Articles for Text Summarization \$\$ Encoder-Decoder  
Models for Text Summarization in Keras  
Word Embeddings

*WhatareWordEmbeddings?*

How to Develop Word Embeddings in Python with Gensim \$\$ How to Use  
Word Embedding Layers for Deep Learning with Keras  
Photo Captioning

*HowtoAutomaticallyGenerateTextualDescriptionsforPhotographswithDeepLearning*

A Gentle Introduction to Deep Learning Caption Generation Models \$\$ How  
to Develop a Deep Learning Photo Caption Generator from Scratch  
Text Translation

*AGentleIntroductiontoNeuralMachineTranslation*

How to Configure an Encoder-Decoder Model for Neural Machine Transla-  
tion \$\$ How to Develop a Neural Machine Translation System from Scratch  
Deep Learning for Time Series Forecasting

Deep learning neural networks are able to automatically learn arbitrary  
complex mappings from inputs to outputs and support multiple inputs and  
outputs.

Methods such as MLPs, CNNs, and LSTMs offer a lot of promise for  
time series forecasting.

Heres how to get started with deep learning for time series forecasting:

\$\$ Step 1: Discover the promise (and limitations) of deep learning for  
time series.

*ThePromiseofRecurrentNeuralNetworksforTimeSeriesForecasting*

On the Suitability of Long Short-Term Memory Networks for Time Series  
Forecasting \$\$ Results From Comparing Classical and Machine Learning  
Methods for Time Series Forecasting

\$\$ Step 2: Discover how to develop robust baseline and defensible fore-  
casting models.

### *Taxonomy of Time Series Forecasting Problems*

How to Develop a Skillful Machine Learning Time Series Forecasting Model

\$\$ Step 3: Discover how to build deep learning models for time series forecasting.

### *How to Get Started with Deep Learning for Time Series Forecasting (7-Day Mini-Course)*

Deep Learning for Time Series Forecasting (my book)

You can see all deep learning for time series forecasting posts here. Below is a selection of some of the most popular tutorials.

Forecast Trends and Seasonality (univariate)

### *Grid Search SARIMA Models for Time Series Forecasting*

Grid Search Exponential Smoothing for Time Series Forecasting \$\$ Develop

Deep Learning Models for Univariate Forecasting

Human Activity Recognition (multivariate classification)

### *How to Model Human Activity From Smartphone Data*

How to Develop CNN Models for Human Activity Recognition \$\$ How to

Develop RNN Models for Human Activity Recognition

Forecast Electricity Usage (multivariate, multi-step)

### *How to Load and Explore Household Electricity Usage Data*

Multi-step Time Series Forecasting with Machine Learning \$\$ How to De-

velop CNNs for Multi-Step Time Series Forecasting

Models Types

### *How to Develop MLPs for Time Series Forecasting*

How to Develop CNNs for Time Series Forecasting \$\$ How to Develop

LSTMs for Time Series Forecasting

Time Series Case Studies

### *Indoor Movement Time Series Classification*

Probabilistic Forecasting Model to Predict Air Pollution Days

### *Predict Room Occupancy Based on Environmental Factors*

Predict Whether Eyes are Open or Closed Using Brain Waves  
Forecast Air Pollution (multivariate, multi-step)

*Load, Visualize, and Explore a Air Pollution Forecasting*

Develop Baseline Forecasts for Air Pollution Forecasting

*Develop Autoregressive Models for Air Pollution Forecasting*

Develop Machine Learning Models for Air Pollution Forecasting

Need More Help?

Im here to help you become awesome at applied machine learning.

If you still have questions and need help, you have some options:

\$\$ Ebooks: I sell a catalog of Ebooks that show you how to get results with machine learning, fast.

\$\$ Machine Learning Mastery EBook Catalog

\$\$ Blog: I write a lot about applied machine learning on the blog, try the search feature.

\$\$ Machine Learning Mastery Blog

\$\$ Frequently Asked Questions: The most common questions I get and their answers

\$\$ Machine Learning Mastery FAQ

\$\$ Contact: You can contact me with your question, but one question at a time please.

\$\$ Machine Learning Mastery Contact

© 2018 Machine Learning Mastery. All Rights Reserved.

## 4 experimental0

chrome history suggestions based on a neural network read to get ideas for projects: <https://machinelearningmastery.com/self-study-machine-learning-projects/>  
maintain a ml daily blog? how to get computers to talk to each other using socket programming use free time with phone to study google maps of Bangalore like guttapalli buy toothbrush, bigger bulb for room sabji kitne ki hai app

## 5 How Do I Get Started?

### 5.1 articles-read, worth rereading

*Applied Machine Learning Process*

Intermediate: Python Ecosystem.

\$\$ Step 4: Practice on Datasets. Select datasets to work on and practice the process.

### *PracticeMachineLearningwithSmallIn – MemoryDatasets*

Tour of Real-World Machine Learning Problems \$\$ Work on Machine Learning Problems That Matter To You

\$\$ Step 5: Build a Portfolio. Gather results and demonstrate your skills.

### *BuildaMachineLearningPortfolio*

Get Paid To Apply Machine Learning \$\$ Machine Learning For Money

For more on this top-down approach, see:

### *TheMachineLearningMasteryMethod*

Machine Learning for Programmers

Many of my students have used this approach to go on and do well in Kaggle competitions and get jobs as Machine Learning Engineers and Data Scientists.

## **5.2 notes**

### **5.2.1 intro**

skit-learn is higher level than numpy & scipy machine learning is a subset of artificial intelligence artificial learning is a more consistent name for machine learning

### **5.2.2 some key word definitions:**

Model: A machine learning model can be a mathematical representation of a real-world process. To generate a machine learning model you will need to provide training data to a machine learning algorithm to learn from.

Algorithm: Machine Learning algorithm is the hypothesis set that is taken at the beginning before the training starts with real-world data. When we say Linear Regression algorithm, it means a set of functions that define similar characteristics as defined by Linear Regression and from those set of functions we will choose one function that fits the most by the training data.

Training: While training for machine learning, you pass an algorithm with training data. The learning algorithm finds patterns in the training

data such that the input parameters correspond to the target. The output of the training process is a machine learning model which you can then use to make predictions. This process is also called learning.

**Regression:** Regression techniques are used when the output is real-valued based on continuous variables. For example, any time series data. This technique involves fitting a line.

**Classification:** In classification, you will need to categorize data into predefined classes. For example, an email can either be spam or not spam.

**Target:** The target is whatever the output of the input variables. It could be the individual classes that the input variables maybe mapped to in case of a classification problem or the output value range in a regression problem. If the training set is considered then the target is the training output values that will be considered.

**Feature:** Features are individual independent variables that act as the input in your system. Prediction models use features to make predictions. New features can also be obtained from old features using a method known as feature engineering. More simply, you can consider one column of your data set to be one feature. Sometimes these are also called attributes. And the number of features are called dimensions.

**Label:** Labels are the final output. You can also consider the output classes to be the labels. When data scientists speak of labeled data, they mean groups of samples that have been tagged to one or more labels.

**Overfitting:** An important consideration in machine learning is how well the approximation of the target function that has been trained using training data, generalizes to new data. Generalization works best if the signal or the sample that is used as the training data has a high signal to noise ratio. If that is not the case, generalization would be poor and we will not get good predictions. A model is overfitting if it fits the training data too well and there is a poor generalization of new data.

**Regularization:** Regularization is the method to estimate a preferred complexity of the machine learning model so that the model generalizes and the over-fit/under-fit problem is avoided. This is done by adding a penalty on the different parameters of the model thereby reducing the freedom of the model.

**Parameter and Hyper-Parameter:** Parameters are configuration variables that can be thought to be internal to the model as they can be estimated from the training data. Algorithms have mechanisms to optimize parameters. On the other hand, hyperparameters cannot be estimated from the training data. Hyperparameters of a model are set and tuned depending on a combination of some heuristics and the experience and domain knowledge of the data

scientist.

6 <https://machinelearningmastery.com/python-machine-learning-mini-14Lessons>

6.1 pre requisite <https://machinelearningmastery.com/gentle-introduction-to-the->

**Bias Error** Bias are the simplifying assumptions made by a model to make the target function easier to learn. Low Bias: Suggests less assumptions about the form of the target function. High-Bias: Suggests more assumptions about the form of the target function.

Examples of low-bias machine learning algorithms include: Decision Trees, k-Nearest Neighbors and Support Vector Machines. Examples of high-bias machine learning algorithms include: Linear Regression, Linear Discriminant Analysis and Logistic Regression.

**Variance Error** Variance is the amount that the estimate of the target function will change if different training data was used.

Examples of low-variance machine learning algorithms include: Linear Regression, Linear Discriminant Analysis and Logistic Regression. Examples of high-variance machine learning algorithms include: Decision Trees, k-Nearest Neighbors and Support Vector Machines.

Fig 1. bulls-eye visualise <http://scott.fortmann-roe.com/docs/BiasVariance.html>

## 6.2 1 - install

6.2.1 next time try using this tutorial <https://sourabhbajaj.com/mac-setup/Python/numpy.html>

6.2.2 make a new virtualenv

`pwd`

Use `:session:` property to speed up org-babel when possible to use the same session

```
source ~/.bashrc
#mkvirtualenv mlm
workon mlm
which python
(pyvenv-workon "mlm2")
```

### 6.2.3 install <https://stackoverflow.com/questions/26319762/how-to-install-scipy-stack-w>

pip install numpy brew install gcc pip install scipy brew install freetype pip install matplotlib pip install nose pip install pandas pip install sympy pip install ipython[all] brew install pyqt brew install qt brew install sip #after this edit the 2 scripts

### 6.2.4 check if properly isntalled using `__version__` after import

using python snippets inside orgmode <https://orgmode.org/worg/org-contrib/babel/languages/ob-doc-python.html> installed python-mode from package-list-packages for emacs

```
(pyvenv-workon "mlm")
```

Fixed some errors using: pip install nose pyparsing python-dateutil pep8

```
import sys
import scipy
import numpy
import matplotlib
import pandas
import sklearn

print(f'python: {sys.version}')
print(f'scipy: {scipy.__version__}')
print(f'numpy: {numpy.__version__}')
print(f'matplotlib {matplotlib.__version__}')
print(f'pandas {pandas.__version__}')
print(f'sklearn {sklearn.__version__}')
```

## 6.3 2 - python, pandas, numpy, mathplotlib veeery basics

### 6.3.1 python, also refer in-y-minutes file for future reference

```
if 1>2:
    print("wtf")
else:
    print("ok")

try:
    # Use "raise" to raise an error
    raise IndexError("This is an index error")
```

```

except IndexError as e:
    print("its indexerror")
    pass # Pass is just a no-op. Usually you would do recovery here.
except (TypeError, NameError):
    print("its typeerror or nameerror")
    pass # Multiple exceptions can be handled together, if required.
else: # Optional clause to the try/except block. Must follow all ex
    print("All good!") # Runs only if the code in try raises no exceptions
finally: # Execute under all circumstances
    print("We can clean up resources here")

def accepts_variable_number_of_arguments(*args):
    print(type(args))
    print(args)

accepts_variable_number_of_arguments(1,2,3)

def accepts_variable_number_of_keyword_arguments(**kwargs):
    print(type(kwargs))
    print(kwargs)

accepts_variable_number_of_keyword_arguments(name="anoop", work="code")

```



## 6.3.2 numpy basics

### Python List vs. Array - when to use?

▲ 306 ▼ ★ 110


If you are creating a 1d array, you can implement it as a List, or else use the 'array' module in the standard library. I have always used Lists for 1d arrays.

What is the reason or circumstance where I would want to use the array module instead?

Is it for performance and memory optimization, or am I missing something obvious?

python arrays list

share improve this question edited Jul 13 '15 at 14:56 asked Oct 6 '08 at 20:17

 Corey Goldberg 36.5k ● 22 ● 106 ● 121

add a comment

9 Answers

active oldest votes

▲ 363 ▼ ✓

Basically, Python lists are very flexible and can hold completely heterogeneous, arbitrary data, and they can be appended to very efficiently, in [amortized constant time](#). If you need to shrink and grow your array time-efficiently and without hassle, they are the way to go. But they use **a lot more space than C arrays**.



The `array.array` type, on the other hand, is just a thin wrapper on C arrays. It can hold only homogeneous data, all of the same type, and so it uses only `sizeof(one object) * length` bytes of memory. Mostly, you should use it when you need to expose a C array to an extension or a system call (for example, `ioctl` or `fctl`).

`array.array` is also a reasonable way to represent a **mutable** string in Python 2.x ( `array('B', bytes)` ). However, Python 2.6+ and 3.x offers a mutable byte string as `bytearray`.

However, if you want to do **math** on a homogeneous array of numeric data, then you're much better off using NumPy, which can automatically vectorize operations on complex multi-dimensional arrays.

**To make a long story short:** `array.array` is useful when you need a homogeneous C array of data for reasons *other than doing math*.

share improve this answer edited Oct 23 at 22:32 answered Oct 6 '08 at 23:11

 user2357112 149k ● 12 ● 155 ● 243  Dan Lenski 48.2k ● 9 ● 56 ● 95

numpy official tutorial <https://docs.scipy.org/doc/numpy-1.15.0/user/quickstart.html>

```
import numpy as np
a = np.arange(15).reshape(3,5)
print(a)
a.shape
a.ndim
a.dtype.name
#dir(a)
a.size
```

```

type(a)
b = np.array([6,7,8])
b
type(b)

np.zeros([2,3])
np.arange(15)
np.linspace(0,9, 19)

from numpy import pi
x = np.linspace(0, 2*pi, 5)
np.sin(x)
#2 decimal places
np.around(np.sin(x), decimals=2)

A = np.array([[1,2],[3,4]])
I = np.array([[1,0],[0,1]])
elementwise = A * I
matrix_product = A @ I
print(elementwise, "\n", matrix_product)

a = np.ones(3, dtype=np.int32)
b = np.linspace(0, 1, 3)
c = a + b
print(a,b,c)
c.dtype.name
c*1j
np.ones(1)
my_e = np.exp(np.ones(1))
from numpy import e
e
print(e, my_e)
d = np.exp(c*1j)
d.dtype
# exp, sin etc are called numpy universal functions

# multidimensional array
c = np.array([
    [
        0, 1, 2],

```

```

    ~I[ 10, 12, 13]
        ],
        [
    ~I[100,101,102],
    ~I[110,112,113]
        ]
    ])
c.shape
# Visualize 0 2,2,3 as you traverse from the topmost bracket to the inner ones
for i in c.flat:
    print(i, end=" // ")
print("\n")
id(c) #id is unique identifier of an object in python

```

axis in numpy

file:~/ml/flipshope/screenshots0/Screenshot 2018-12-11 at 6.06.43 PM.png

matplotlib python is not installed as a framework error, solution: <https://stackoverflow.com/questions/34977388/matplotlib-runtimeerror-python-is-not-installed>

Above is a hacky solution I need to switch away from virtualenv & virtualenvwrapper and move to venv entirely Also, reddit recommends to avoid the pyenv or any wrapper around venv && strongly recommends to use venv directly venv ships by default with python >= 3.3 [https://matplotlib.org/faq/osx\\_framework.html](https://matplotlib.org/faq/osx_framework.html) <https://news.ycombinator.com/item?id=18612590> <https://news.ycombinator.com/item?id=18247512> :) andybak 54 days ago [-] In case this scares any new users, I've used nothing more than pip and virtualenv for several years with no issues of note.

```

import numpy as np
import matplotlib.pyplot as plt
#import matplotlib
#matplotlib.use('TkAgg')
#import matplotlib.pyplot as plt

def mandelbrot( h,w, maxit=20 ):
    """Returns an image of the Mandelbrot fractal of size (h,w)."""
    y,x = np.ogrid[ -1.4:1.4:h*1j, -2:0.8:w*1j ]
    c = x+y*1j
    z = c
    divtime = maxit + np.zeros(z.shape, dtype=int)

```

```

        for i in range(maxit):
            ^^Iz = z**2 + c
            ^^Idiverge = z*np.conj(z) > 2**2          # who is diverging
            ^^Idiv_now = diverge & (divtime==maxit)   # who is diverging now
            ^^Idivtime[div_now] = i                  # note when
            ^^Iz[diverge] = 2                         # avoid diverging too much

        return divtime
plt.imshow(mandelbrot(400,400))
plt.show()

import sys
print(sys.path)

```

**real-python tutorial** <https://realpython.com/numpy-array-programming/>

When it comes to computation, there are really three concepts that lend NumPy its power: Vectorization Broadcasting Indexing

```

import numpy as np

arr = np.arange(36).reshape(3,4,3)
arr

"""
visualize0:-

00 01 02 03 04 05 06 07 08 09 10 11
12 13 14 15 16 17 18 19 20 21 22 23
24 25 26 27 28 29 30 31 32 33 34 35

00 01 02
03 04 05
06 07 08
09 10 11

[#3 items
[#4 items
[]
[]
[]

```

```

[]
]...
]
"""
a = np.array([2,3,4])
b = 2
b_broadcasted = np.broadcast(a,b)
print(list(b_broadcasted))
# rest of this tutorial seemed a bit advanced, skip for now, come back later

```

**Note** Pandas is a library built on top of NumPy

todo: switch to jupyter notebook instead of emacs: <https://github.com/millejoh/emacs-ipython-notebook> <http://millejoh.github.io/emacs-ipython-notebook/> <https://www.youtube.com/watch?v=wtVF5cMhBjg> <https://news.ycombinator.com/item?id=9728143> <https://github.com/gregsexton/ob-ipython>

### 6.3.3 matplotlib basics

matplotlib, is written in pure Python and is heavily dependent on NumPy

Matplotlib is conceptually divided into three parts: pylab interface (similar to MATLAB) pylab tutorial, this is the matplotlib.pyplot import Matplotlib frontend or API artist tutorial backends drawing devices or renderers

Lets learn pyplot [https://matplotlib.org/users/pyplot\\_tutorial.html#pyplot-tutorial](https://matplotlib.org/users/pyplot_tutorial.html#pyplot-tutorial)

venv basics to switch from virtualenv

python3 -m venv mlm2 source mlm2/bin/activate

pyvenv-deactivate

```

import matplotlib
#matplotlib.use('MacOSX')
import matplotlib.pyplot as plt

```

```

plt.plot([0, 1, 4, 9, 16])
#plt.show()

```

```

plt.plot([0, 1, 4, 9, 16], 'ro')
#plt.show()

```

```

plt.plot([0.5], 'y.')
plt.show()

```

### 6.3.4 Pandas -basics:

Todo:<https://pandas.pydata.org/pandas-docs/stable/dsintro.html#dsintro>

Todo:Official beginner tutorial: <https://pandas.pydata.org/pandas-docs/stable/10min.html> Todo: Intermediate - Julia Evans - <https://jvns.ca/blog/2013/12/22/cooking-with-pandas/> "take a real dataset or three, play around with it, and learn how to use pandas along the way."

Panda Series & DataFrames: <https://medium.freecodecamp.org/series-and-dataframe-in-pyt>

```
#Series and DataFrames
```

```
import pandas as pd
```

```
x1 = pd.Series([6,3,4,6])
```

```
x = pd.Series([6,3,4,6], index=['a','b','c','d'])
```

```
x
```

```
y = pd.Series(3, index=['a', 'b', 'c', 'd'])
```

```
y
```

```
#DataFrames
```

```
import numpy as np
```

```
dates = pd.date_range('20181201', periods = 8)
```

```
my_narray = np.random.randn(8,3)
```

```
list('ABC')
```

```
df = pd.DataFrame(index = dates, data = my_narray, columns = ['A','B','C'])
```

```
df_absolute = df.apply(abs)
```

So pandas is kinda like an excel sheet0

```
import numpy as np
```

```
np.arange(4)
```

```
ma = np.arange(4).reshape((2,2))
```

```
import pandas
```

```
pandas.DataFrame(ma)
```

```
print(ma[1,0])
```

```
print(p[1])
```

```
print(p[1][0] == ma[1][0])
```

```
print(p.shape)
```

## 6.4 3 - Load csv

<https://realpython.com/python-csv/> <https://github.com/jbrownlee/Datasets>

Work with csv using python's csv module

```
import csv

with open('iris.csv') as csv_file:
    """
    for line in csv_file:
        ^Iprint(line)
    ^Ipass
    """
    csv_reader = csv.reader(csv_file)
    for line in csv_reader:
        ^I#print(line)
    ^Ipass

my_fieldnames = ("sepal_length", "sepal_width", "petal_length", "petal_width", "class")

with open('iris.csv') as csv_file:
    csv_dict_reader = csv.DictReader(csv_file, fieldnames=my_fieldnames)
    for line in csv_dict_reader:
        ^I#print(line)
    ^Ipass

with open('test_writeout.csv', mode='w') as out_file:
    csv_writer = csv.writer(out_file)
    csv_writer.writerow(["row", "1"])
    csv_writer.writerow(["row", "2"])
    #
    csv_dict_writer = csv.DictWriter(out_file, fieldnames = my_fieldnames)
    csv_dict_writer.writeheader()
    csv_dict_writer.writerow({"sepal_length": 1, "sepal_width": 2, "petal_length": 3,
```

Work with csv using numpy

```
import numpy as np

with open("numpy_loadtxt_input.txt") as input_file:
```

```

    """
    for line in input_file:
        print(line)
    """

    my_narray = np.loadtxt(input_file, delimiter=" ")
    print(my_narray)
    print(my_narray.dtype)

```

### Work with csv usign pandas

```

import pandas
df = pandas.read_csv('pandas_read_csv.csv')
print(df)

df2 = pandas.read_csv('pandas_read_csv.csv', parse_dates=['Hire Date'], index_col='Name')
print(df2)

my_col_names = ("Name", "Hired_on", "Salary", "sick_days_remaining")
df3 = pandas.read_csv('pandas_read_csv.csv', header=None, names=my_col_names,
parse_dates=['Hired_on'])
print(df3)

df3.to_csv('pandas_to_csv.csv')

```

### 6.5 4 - use pandas.DataFrame helper functions to describe data with statistics

```

# Scatter Plot Matrix
import matplotlib.pyplot as plt
import pandas

my_col_names = ("num_preg", "plasma_glucose", "blood_pressure", "triceps_skin_thickness")
df = pandas.read_csv("https://raw.githubusercontent.com/jbrownlee/Datasets/master/pima-indi")
#print(df)

from pandas.plotting import scatter_matrix

scatter_matrix(df)
plt.show()

df.corr()

```



## 6.6 5 - Basic Data Visualization

Using pandas and matplotlib together

```
# Scatter Plot Matrix
import matplotlib.pyplot as plt
import pandas

my_fieldnames = ("sepal_length", "sepal_width", "petal_length", "petal_width", "class")
data = pandas.read_csv('iris.csv', names=my_fieldnames)
#print(data)

se = data.loc[data["class"] == "Iris-setosa"]
ve = data.loc[data["class"] == "Iris-versicolor"]
vi = data.loc[data["class"] == "Iris-virginica"]

se.hist()
#ve.hist()
#vi.hist()

from pandas.plotting import scatter_matrix
scatter_matrix(se)

plt.show()
```