Syllabus: Intro to Machine Learning Nanodegree Program

Prerequisites

To optimize your chances of success in this program, we recommend having experience with:

- Intermediate Python programming knowledge, including:
  - At least 40hrs of programming experience
  - Familiarity with data structures like dictionaries and lists
  - Experience with libraries like NumPy and pandas

- Basic knowledge of probability and statistics, including:
  - Experience calculating the probability of an event
  - Familiarity with terms like the mean and variance of a probability distribution

There are a few courses that can help prepare you for this program, depending on the areas you need to address. For example:

- AI Programming with Python Nanodegree Program
- Intro to Programming Nanodegree Program

Contact Info

While going through the program, if you have questions about anything, you can reach us at: machine-support@udacity.com.

Program Overview

This ultimate goal of the Intro to Machine Learning Nanodegree program is to help students learn machine learning techniques such as data transformation and algorithms that can find patterns in data and apply machine learning algorithms to tasks of their own design.

A graduate of this program will be able to:

- Use Python and SQL to access and analyze data from several different data sources.
- Build predictive models using a variety of unsupervised and supervised machine learning techniques.
- Perform feature engineering to improve the performance of machine learning models.
- Optimize, tune, and improve algorithms according to specific metrics like accuracy and speed.
- Compare the performances of learned models using suitable metrics.

This program is comprised of 3 courses and 3 projects. Each project you build will be an opportunity to demonstrate what you've learned in the lessons. **Your completed projects will become part of a career portfolio that will demonstrate to potential employers that you have skills in data analysis and feature engineering, machine learning algorithms, and training and evaluating models.**

Estimated Length of Program: 3 months
Frequency of Classes: Self-paced

Projects Overview

One of our main goals at Udacity is to help you **create a job-ready portfolio of completed projects**. Building a project is one of the best ways to test the skills you've acquired and to demonstrate your newfound abilities to future employers or colleagues. Throughout this Nanodegree program, you'll have the opportunity to prove your skills by building the following projects:

- **Finding Donors for CharityML**
    - Apply supervised learning techniques on data collected for the US census to help CharityML (a fictitious charity organization) identify groups of people that are most likely to donate to their cause.
- **Create Your Own Image Classifier**
    - Define and train a neural network in PyTorch that learns to classify images; going from image data exploration to network training and evaluation.
- **Identify Customer Segments with Arvato**
    - Study a real dataset of customers for a company, and apply several unsupervised learning techniques in order to segment customers into similar groups and extract information that may be used for marketing or product improvement.

In the sections below, you'll find detailed descriptions of each project along with the course material that presents the skills required to complete the project.

Project: Find Donors for CharityML

Project Description:

CharityML is a fictitious charity organization located in the heart of Silicon Valley that was established to provide financial support for people eager to learn machine learning. To expand their potential donor base, CharityML has decided to send letters to residents of California, but to only those most likely to donate to the charity. Your goal will be to evaluate and optimize several different supervised learning algorithms to determine which algorithm will provide the highest donation yield while under some marketing constraints.

Key Skills Demonstrated:

- Supervised learning
- Model evaluation and comparison
- Tuning models according to constraints

Supporting Lesson Content: Supervised Learning

Lesson Title	Learning Outcomes
Regression	➔ Learn the difference between Regression and Classification ➔ Train a Linear Regression model to predict values ➔ Learn to predict states using Logistic Regression
Perceptron Algorithms	➔ Learn the definition of a perceptron as a building block for neural networks, and the perceptron algorithm for classification.
Decision Trees	➔ Train Decision Trees to predict states ➔ Use Entropy to build decision trees, recursively
Naive Bayes'	➔ Learn Bayes' rule, and apply it to predict cases of spam messages using the Naive Bayes algorithm ➔ Train models using Bayesian Learning ➔ Complete an exercise that uses Bayesian Learning for natural language processing
Support Vector Machines	➔ Learn to train a Support Vector Machines to separate data, linearly ➔ Use Kernel Methods in order to train SVMs on data that is not linearly separable
Ensemble of Learners	➔ Enhance traditional algorithms via boosting ➔ Learn and apply Random Forest algorithms

	➔ Use AdaBoost and evaluate the performance of boosted models
Evaluation Metrics	➔ Learn about different metrics to measure model success ➔ Calculate accuracy, precision, and recall to measure the performance of your models.
Training and Tuning Models	➔ Train and test models with Scikit-learn. ➔ Choose the best model using evaluation techniques like cross-validation and grid search.

Project: Create an Image Classifier

Project Description:

Implementing an image classification application using a deep neural network. This application will train a deep learning model on a dataset of images. It will then use the trained model to classify new images. You will develop your code in a Jupyter notebook to ensure your implementation works well.

Key Skills Demonstrated:

- PyTorch and neural networks
- Model validation and evaluation

Supporting Lesson Content: Neural Networks

Lesson Tit	Learning Outcomes
Introduction to Neural Networks	➔ Learn the foundations of deep learning and neural networks. ➔ Implement gradient descent and backpropagation in Python.
Implementing Gradient Descent	➔ Implement gradient descent using NumPy matrix multiplication.
Training Neural Networks	➔ Learn several techniques to effectively train a neural network ➔ Prevent overfitting of training data and learn best practices for minimizing the error of a network.
Deep Learning with PyTorch	➔ Learn how to use PyTorch for building deep learning models.

Project: Creating Customer Segments

Project Description:

In this project, you will apply unsupervised learning techniques on product spending data collected for customers of a wholesale distributor in Lisbon, Portugal to identify customer segments hidden in the data. You will first explore and pre-process the data by scaling each product category and then identifying (and removing) unwanted outliers. With the cleaned customer spending data, you will apply PCA transformations to the data and implement clustering algorithms to segment the transformed customer data. Finally, you will compare the segmentation found with an additional labeling and consider ways this information could assist the wholesale distributor with future service changes.

Key Skills Demonstrated:

- Data cleaning
- Dimensionality reduction with PCA
- Unsupervised clustering

Supporting Lesson Content: Unsupervised Learning

Lesson Title	Learning Outcomes
Clustering	➔ Learn the basics of clustering data ➔ Cluster data with the K-means algorithm
Hierarchical and Density-Based Clustering	➔ Cluster data with Single Linkage Clustering ➔ Cluster data with DBSCAN, a clustering method that captures the insight that clusters are dense group of points
Gaussian Mixture Models	➔ Cluster data with Gaussian Mixture Models ➔ Optimize Gaussian Mixture Models with and Expectation Maximization
Dimensionality Reduction	➔ Reduce the dimensionality of the data using Principal Component Analysis and Independent Component Analysis