

1. Record Nr.	UNINA9911019987303321
Autore	Vasques Xavier
Titolo	Machine Learning Theory and Applications : Hands-On Use Cases with Python on Classical and Quantum Machines
Pubbl/distr/stampa	Newark : , : John Wiley & Sons, Incorporated, , 2024 ©2024
ISBN	9781394220649 1394220642 9781394220632 1394220634
Edizione	[1st ed.]
Descrizione fisica	1 online resource (510 pages)
Disciplina	006.3/1
Soggetti	Machine learning Quantum computing Python (Computer program language)
Lingua di pubblicazione	Inglese
Formato	Materiale a stampa
Livello bibliografico	Monografia
Nota di contenuto	Cover -- Title Page -- Copyright Page -- Dedication Page -- Contents -- Foreword -- Acknowledgments -- General Introduction -- Chapter 1 Concepts, Libraries, and Essential Tools in Machine Learning and Deep Learning -- 1.1 Learning Styles for Machine Learning -- 1.1.1 Supervised Learning -- 1.1.1.1 Overfitting and Underfitting -- 1.1.1.2 K-Folds Cross-Validation -- 1.1.1.3 Train/Test Split -- 1.1.1.4 Confusion Matrix -- 1.1.1.5 Loss Functions -- 1.1.2 Unsupervised Learning -- 1.1.3 Semi-Supervised Learning -- 1.1.4 Reinforcement Learning -- 1.2 Essential Python Tools for Machine Learning -- 1.2.1 Data Manipulation with Python -- 1.2.2 Python Machine Learning Libraries -- 1.2.2.1 Scikit-learn -- 1.2.2.2 TensorFlow -- 1.2.2.3 Keras -- 1.2.2.4 PyTorch -- 1.2.3 Jupyter Notebook and JupyterLab -- 1.3 HephAlstos for Running Machine Learning on CPUs, GPUs, and QPUs -- 1.3.1 Installation -- 1.3.2 HephAlstos Function -- 1.4 Where to Find the Datasets and Code Examples -- Further Reading -- Chapter 2 Feature Engineering Techniques in Machine Learning -- 2.1 Feature Rescaling: Structured Continuous Numeric Data -- 2.1.1 Data

Transformation -- 2.1.1.1 StandardScaler -- 2.1.1.2 MinMaxScaler -- 2.1.1.3 MaxAbsScaler -- 2.1.1.4 RobustScaler -- 2.1.1.5 Normalizer: Unit Vector Normalization -- 2.1.1.6 Other Options -- 2.1.1.7 Transformation to Improve Normal Distribution -- 2.1.1.8 Quantile Transformation -- 2.1.2 Example: Rescaling Applied to an SVM Model -- 2.2 Strategies to Work with Categorical (Discrete) Data -- 2.2.1 Ordinal Encoding -- 2.2.2 One-Hot Encoding -- 2.2.3 Label Encoding -- 2.2.4 Helmert Encoding -- 2.2.5 Binary Encoding -- 2.2.6 Frequency Encoding -- 2.2.7 Mean Encoding -- 2.2.8 Sum Encoding -- 2.2.9 Weight of Evidence Encoding -- 2.2.10 Probability Ratio Encoding -- 2.2.11 Hashing Encoding -- 2.2.12 Backward Difference Encoding -- 2.2.13 Leave-One-Out Encoding -- 2.2.14 James-Stein Encoding -- 2.2.15 M-Estimator Encoding -- 2.2.16 Using HephAlstos to Encode Categorical Data -- 2.3 Time-Related Features Engineering -- 2.3.1 Date-Related Features -- 2.3.2 Lag Variables -- 2.3.3 Rolling Window Feature -- 2.3.4 Expanding Window Feature -- 2.3.5 Understanding Time Series Data in Context -- 2.4 Handling Missing Values in Machine Learning -- 2.4.1 Row or Column Removal -- 2.4.2 Statistical Imputation: Mean, Median, and Mode -- 2.4.3 Linear Interpolation -- 2.4.4 Multivariate Imputation by Chained Equation Imputation -- 2.4.5 KNN Imputation -- 2.5 Feature Extraction and Selection -- 2.5.1 Feature Extraction -- 2.5.1.1 Principal Component Analysis -- 2.5.1.2 Independent Component Analysis -- 2.5.1.3 Linear Discriminant Analysis -- 2.5.1.4 Locally Linear Embedding -- 2.5.1.5 The t-Distributed Stochastic Neighbor Embedding Technique -- 2.5.1.6 More Manifold Learning Techniques -- 2.5.1.7 Feature Extraction with HephAlstos -- 2.5.2 Feature Selection -- 2.5.2.1 Filter Methods -- 2.5.2.2 Wrapper Methods -- 2.5.2.3 Embedded Methods -- 2.5.2.4 Feature Importance Using Graphics Processing Units (GPUs) -- 2.5.2.5 Feature Selection Using HephAlstos -- Further Reading -- Chapter 3 Machine Learning Algorithms -- 3.1 Linear Regression -- 3.1.1 The Math -- 3.1.2 Gradient Descent to Optimize the Cost Function -- 3.1.3 Implementation of Linear Regression -- 3.1.3.1 Univariate Linear Regression -- 3.1.3.2 Multiple Linear Regression: Predicting Water Temperature -- 3.2 Logistic Regression -- 3.2.1 Binary Logistic Regression -- 3.2.1.1 Cost Function -- 3.2.1.2 Gradient Descent -- 3.2.2 Multinomial Logistic Regression -- 3.2.3 Multinomial Logistic Regression Applied to Fashion MNIST -- 3.2.3.1 Logistic Regression with scikit-learn -- 3.2.3.2 Logistic Regression with Keras on TensorFlow. -- 3.2.4 Binary Logistic Regression with Keras on TensorFlow -- 3.3 Support Vector Machine -- 3.3.1 Linearly Separable Data -- 3.3.2 Not Fully Linearly Separable Data -- 3.3.3 Nonlinear SVMs -- 3.3.4 SVMs for Regression -- 3.3.5 Application of SVMs -- 3.3.5.1 SVM Using scikit-learn for Classification -- 3.3.5.2 SVM Using scikit-learn for Regression -- 3.4 Artificial Neural Networks -- 3.4.1 Multilayer Perceptron -- 3.4.2 Estimation of the Parameters -- 3.4.2.1 Loss Functions -- 3.4.2.2 Backpropagation: Binary Classification -- 3.4.2.3 Backpropagation: Multi-class Classification -- 3.4.3 Convolutional Neural Networks -- 3.4.4 Recurrent Neural Network -- 3.4.5 Application of MLP Neural Networks -- 3.4.6 Application of RNNs: LST Memory -- 3.4.7 Building a CNN -- 3.5 Many More Algorithms to Explore -- 3.6 Unsupervised Machine Learning Algorithms -- 3.6.1 Clustering -- 3.6.1.1 K-means -- 3.6.1.2 Mini-batch K-means -- 3.6.1.3 Mean Shift -- 3.6.1.4 Affinity Propagation -- 3.6.1.5 Density-based Spatial Clustering of Applications with Noise -- 3.7 Machine Learning Algorithms with HephAlstos -- References -- Further Reading -- Chapter 4 Natural Language Processing -- 4.1 Classifying Messages

as Spam or Ham -- 4.2 Sentiment Analysis -- 4.3 Bidirectional Encoder Representations from Transformers -- 4.4 BERT's Functionality -- 4.5 Installing and Training BERT for Binary Text Classification Using TensorFlow -- 4.6 Utilizing BERT for Text Summarization -- 4.7 Utilizing BERT for Question Answering -- Further Reading -- Chapter 5 Machine Learning Algorithms in Quantum Computing -- 5.1 Quantum Machine Learning -- 5.2 Quantum Kernel Machine Learning -- 5.3 Quantum Kernel Training -- 5.4 Pegasos QSVC: Binary Classification -- 5.5 Quantum Neural Networks -- 5.5.1 Binary Classification with EstimatorQNN -- 5.5.2 Classification with a SamplerQNN. 5.5.3 Classification with Variational Quantum Classifier -- 5.5.4 Regression -- 5.6 Quantum Generative Adversarial Network -- 5.7 Quantum Algorithms with HephAistos -- References -- Further Reading -- Chapter 6 Machine Learning in Production -- 6.1 Why Use Docker Containers for Machine Learning? -- 6.1.1 First Things First: The Microservices -- 6.1.2 Containerization -- 6.1.3 Docker and Machine Learning: Resolving the "It Works in My Machine" Problem -- 6.1.4 Quick Install and First Use of Docker -- 6.1.4.1 Install Docker -- 6.1.4.2 Using Docker from the Command Line -- 6.1.5 Dockerfile -- 6.1.6 Build and Run a Docker Container for Your Machine Learning Model -- 6.2 Machine Learning Prediction in Real Time Using Docker and Python REST APIs with Flask -- 6.2.1 Flask-RESTful APIs -- 6.2.2 Machine Learning Models -- 6.2.3 Docker Image for the Online Inference -- 6.2.4 Running Docker Online Inference -- 6.3 From DevOps to MLOPS: Integrate Machine Learning Models Using Jenkins and Docker -- 6.3.1 Jenkins Installation -- 6.3.2 Scenario Implementation -- 6.4 Machine Learning with Docker and Kubernetes: Install a Cluster from Scratch -- 6.4.1 Kubernetes Vocabulary -- 6.4.2 Kubernetes Quick Install -- 6.4.3 Install a Kubernetes Cluster -- 6.4.4 Kubernetes: Initialization and Internal Network -- 6.5 Machine Learning with Docker and Kubernetes: Training Models -- 6.5.1 Kubernetes Jobs: Model Training and Batch Inference -- 6.5.2 Create and Prepare the Virtual Machines -- 6.5.3 Kubeadm Installation -- 6.5.4 Create a Kubernetes Cluster -- 6.5.5 Containerize our Python Application that Trains Models -- 6.5.6 Create Configuration Files for Kubernetes -- 6.5.7 Commands to Delete the Cluster -- 6.6 Machine Learning with Docker and Kubernetes: Batch Inference -- 6.6.1 Create Configuration Files for Kubernetes. 6.7 Machine Learning Prediction in Real Time Using Docker, Python Rest APIs with Flask, and Kubernetes: Online Inference -- 6.7.1 Flask-RESTful APIs -- 6.7.2 Machine Learning Models -- 6.7.3 Docker Image for Online Inference -- 6.7.4 Running Docker Online Inference -- 6.7.5 Create and Prepare the Virtual Machines -- 6.7.6 Kubeadm Installation -- 6.7.7 Create a Kubernetes Cluster -- 6.7.8 Deploying the Containerized Machine Learning Model to Kubernetes -- 6.8 A Machine Learning Application that Deploys to the IBM Cloud Kubernetes Service: Python, Docker, Kubernetes -- 6.8.1 Create Kubernetes Service on IBM Cloud -- 6.8.2 Containerization of a Machine Learning Application -- 6.8.3 Push the Image to the IBM Cloud Registry -- 6.8.4 Deploy the Application to Kubernetes -- 6.9 Red Hat OpenShift to Develop and Deploy Enterprise ML/DL Applications -- 6.9.1 What is OpenShift? -- 6.9.2 What Is the Difference Between OpenShift and Kubernetes? -- 6.9.3 Why Red Hat OpenShift for ML/DL? To Build a Production-Ready ML/DL Environment -- 6.10 Deploying a Machine Learning Model as an API on the Red Hat OpenShift Container Platform: From Source Code in a GitHub Repository with Flask, Scikit-Learn, and Docker -- 6.10.1 Create an OpenShift Cluster Instance -- 6.10.1.1 Deploying an Application from Source Code in a GitHub Repository -- Further

Sommario/riassunto

"Machine learning (ML) and quantum computing are two technologies that have the potential to allow us to solve complex, previously impossible problems and help speed up areas such as model training or pattern recognition. The future of computing will certainly be comprised of classical, biologically inspired, and quantum computing. The intersection between quantum computing and AI/ML has received considerable attention in recent years and has enabled the development of quantum machine learning algorithms such as quantum-enhanced Support Vector Machines (QSVMs), QSVM multiclass classification, variational quantum classifiers or quantum generative adversarial networks (qGANs)."--
