1. **Python for ML/AI**
    1.1. Why Python?
    1.2. Setup
        1.2.1. Install Python.
        1.2.2. Installing packages: numpy, pandas, scipy, matplotlib, seaborn, sklearn)
        1.2.3. iPython setup.
    1.3. Introduction
        1.3.1. Keywords and Identifiers
        1.3.2. Statements, Indentation and Comments
        1.3.3. Variables and Datatypes
        1.3.4. Input and Output
        1.3.5. Operators
    1.4. Flow Control
        1.4.1. If...else
        1.4.2. while loop
        1.4.3. for loop
        1.4.4. break and continue
    1.5. Data Structures
        1.5.1. Lists
        1.5.2. Tuples
        1.5.3. Dictionary
        1.5.4. Strings
        1.5.5. Sets
    1.6. Functions
        1.6.1. Introduction
        1.6.2. Types of functions
        1.6.3. Function Arguments
        1.6.4. Recursive Functions
        1.6.5. Lambda Functions
        1.6.6. Modules
        1.6.7. Packages
    1.7. File Handling
    1.8. Exception Handling
    1.9. Debugging Python
    1.10. NumPy
        1.10.1. Introduction to NumPy.
        1.10.2. Numerical operations.
    1.11. Matplotlib
    1.12. Pandas
        1.12.1. Getting started with pandas

1.12.2. Data Frame Basics
1.12.3. Key Operations on Data Frames.
1.13. Computational Complexity: an Introduction
1.13.1. Space and Time Complexity: Find largest number in a list
1.13.2. Binary search
1.13.3. Find elements common in two lists.
1.13.4. Find elements common in two lists using a Hashtable/Dict
1.13.5. Further reading about Computational Complexity .

2. Plotting for exploratory data analysis (EDA)
2.1. Iris dataset
2.1.1. Data-point, vector, observation
2.1.2. Dataset
2.1.3. Input variables/features/dimensions/independent variable
2.1.4. Output Variable/Class Label/ Response Label/ dependent variable
2.1.5. Objective: Classification.
2.2. Scatter-plot: 2D, 3D.
2.3. Pair plots.
2.4. PDF, CDF, Univariate analysis.
2.4.1. Histogram and PDF
2.4.2. Univariate analysis using PDFs.
2.4.3. Cumulative distribution function (CDF)
2.5. Mean , Variance, Std-dev
2.6. Median, Percentiles, Quantiles, IQR, MAD and Outliers.
2.7. Box-plot with whiskers
2.8. Violin plots.
2.9. Summarizing plots.
2.10. Univariate, Bivariate and Multivariate analysis.
2.11. Multivariate probability density, contour plot.
2.12. Exercise: Perform EDA on Haberman dataset.

3. Probability and Statistics
3.1. Introduction to Probability and Stats
3.1.1. Why learn it?
3.1.2. P(X=x1) , Dice and coin example
3.1.3. Random variables: discrete and continuous.
3.1.4. Outliers (or) extreme points.
3.1.5. Population & Sample.
3.2. Gaussian/Normal Distribution
3.2.1. Examples: Heights and weights.
3.2.2. Why learn about distributions.
3.2.3. Mu, sigma: Parameters
3.2.4. PDF (iris dataset)
3.2.5. CDF

- 3.2.6. 1-std-dev, 2-std-dev, 3-std-dev range.
- 3.2.7. Symmetric distribution, Skewness and Kurtosis
- 3.2.8. Standard normal variate (z) and standardization.
- 3.2.9. Kernel density estimation.
- 3.2.10. Sampling distribution & Central Limit theorem.
- 3.2.11. Q-Q Plot: Is a given random variable Gaussian distributed?

3.3. Uniform Distribution and random number generators
- 3.3.1. Discrete and Continuous Uniform distributions.
- 3.3.2. How to randomly sample data points. [UniformDisb.ipynb]

3.4. Bernoulli and Binomial distribution

3.5. Log-normal and power law distribution:
- 3.5.1. Log-normal: CDF, PDF, Examples.
- 3.5.2. Power-law & Pareto distributions: PDF, examples
- 3.5.3. Converting power law distributions to normal: Box-Cox/Power transform.

3.6. Correlation
- 3.6.1. Co-variance
- 3.6.2. Pearson Correlation Coefficient
- 3.6.3. Spearman Rank Correlation Coefficient
- 3.6.4. Correlation vs Causation

3.7. Confidence Intervals
- 3.7.1. Confidence Interval vs Point estimate.
- 3.7.2. Computing confidence-interval given a distribution.
- 3.7.3. For mean of a random variable
  - 3.7.3.1. Known Standard-deviation: using CLT
  - 3.7.3.2. Unknown Standard-deviation: using t-distribution
- 3.7.4. Confidence Interval using empirical bootstrap [BootstrapCI.ipynb]

3.8. Hypothesis testing
- 3.8.1. Hypothesis Testing methodology, Null-hypothesis, test-statistic, p-value.
- 3.8.2. Resampling and permutation test.
- 3.8.3. K-S Test for similarity of two distributions.
- 3.8.4. Code Snippet [KSTest.ipynb]

4. Linear Algebra

4.1. Why learn it ?

4.2. Fundamentals
- 4.2.1. Point/Vector (2-D, 3-D, n-D)
- 4.2.2. Dot product and angle between 2 vectors.
- 4.2.3. Projection, unit vector
- 4.2.4. Equation of a line (2-D), plane(3-D) and hyperplane (n-D)
- 4.2.5. Distance of a point from a plane/hyperplane, half-spaces
- 4.2.6. Equation of a circle (2-D), sphere (3-D) and hypersphere (n-D)
- 4.2.7. Equation of an ellipse (2-D), ellipsoid (3-D) and hyperellipsoid (n-D)
- 4.2.8. Square, Rectangle, Hyper-cube and Hyper-cuboid..

5. Dimensionality reduction and Visualization:

- 5.1. What is dimensionality reduction?
- 5.2. Data representation and pre-processing
  - 5.2.1. Row vector, Column vector: Iris dataset example.
  - 5.2.2. Represent a dataset: D= {x_i, y_i}
  - 5.2.3. Represent a dataset as a Matrix.
  - 5.2.4. Data preprocessing: Column Normalization
  - 5.2.5. Mean of a data matrix.
  - 5.2.6. Data preprocessing: Column Standardization
  - 5.2.7. Co-variance of a Data Matrix.
- 5.3. MNIST dataset (784 dimensional)
  - 5.3.1. Explanation of the dataset.
  - 5.3.2. Code to load this dataset.
- 5.4. Principal Component Analysis.
  - 5.4.1. Why learn it.
  - 5.4.2. Geometric intuition.
  - 5.4.3. Mathematical objective function.
  - 5.4.4. Alternative formulation of PCA: distance minimization
  - 5.4.5. Eigenvalues and eigenvectors.
  - 5.4.6. PCA for dimensionality reduction and visualization.
  - 5.4.7. Visualize MNIST dataset.
  - 5.4.8. Limitations of PCA
  - 5.4.9. Code example.
  - 5.4.10. PCA for dimensionality reduction (not-visualization)
- 5.5. T-distributed stochastic neighborhood embedding (t-SNE)
  - 5.5.1. What is t-SNE?
  - 5.5.2. Neighborhood of a point, Embedding.
  - 5.5.3. Geometric intuition.
  - 5.5.4. Crowding problem.
  - 5.5.5. How to apply t-SNE and interpret its output (distill.pub)
  - 5.5.6. t-SNE on MNIST.
  - 5.5.7. Code example.

6. Real world problem: Predict sentiment polarity given product reviews on Amazon.

- 6.1. Exploratory Data Analysis.
  - 6.1.1. Dataset overview: Amazon Fine Food reviews
  - 6.1.2. Data Cleaning: Deduplication.
- 6.2. Featurizations: convert text to numeric vectors.
  - 6.2.1. Why convert text to a vector?
  - 6.2.2. Bag of Words (BoW)
  - 6.2.3. Text Preprocessing: Stemming, Stop-word removal, Tokenization, Lemmatization.
  - 6.2.4. uni-gram, bi-gram, n-grams.
  - 6.2.5. tf-idf (term frequency- inverse document frequency)
  - [6.2.5 a] [New Video] Why use log in IDF?

6.2.6. Word2Vec.
6.2.7. Avg-Word2Vec, tf-idf weighted Word2Vec
6.3. Code samples
6.3.1. Bag of Words.
6.3.2. Text Preprocessing
6.3.3. Bi-Grams and n-grams.
6.3.4. TF-IDF
6.3.5. Word2Vec
6.3.6. Avg-Word2Vec and TFIDF-Word2Vec
6.4. Exercise: t-SNE visualization of Amazon reviews with polarity based color-coding

7. Classification and Regression Models: K-Nearest Neighbors
7.1. Foundations
7.1.1. How “Classification” works?
7.1.2. Data matrix notation.
7.1.3. Classification vs Regression (examples)
7.2. K-Nearest Neighbors
7.2.1. Geometric intuition with a toy example.
7.2.2. Failure cases.
7.2.3. Distance measures: Euclidean(L2) , Manhattan(L1), Minkowski, Hamming
7.2.4. Cosine Distance & Cosine Similarity
7.2.5. How to measure the effectiveness of k-NN?
7.2.6. Simple implementation:
7.2.6.1. Test/Evaluation time and space complexity.
7.2.6.2. Limitations.
7.2.7. Determining the right “k”
7.2.7.1. Decision surface for K-NN as K changes.
7.2.7.2. Overfitting and Underfitting.
7.2.7.3. Need for Cross validation.
7.2.7.4. K-fold cross validation.
[NEW]8.2.7.4 a Visualizing train, validation and test datasets
7.2.7.5. How to determine overfitting and underfitting?
7.2.7.6. Time based splitting
7.2.8. k-NN for regression.
7.2.9. Weighted k-NN
7.2.10. Voronoi diagram.
7.2.11. kd-tree based k-NN:
7.2.11.1. Binary search tree
7.2.11.2. How to build a kd-tree.
7.2.11.3. Find nearest neighbors using kd-tree
7.2.11.4. Limitations.
7.2.11.5. Extensions.

- 7.2.12. Locality sensitive Hashing (LSH)
  - 7.2.12.1. Hashing vs LSH.
  - 7.2.12.2. LSH for cosine similarity
  - 7.2.12.3. LSH for euclidean distance.
- 7.2.13. Probabilistic class label
- 7.2.14. Code Samples for K-NN
  - 7.2.14.1. Decision boundary. [./knn/knn.ipynb and knn folder]
  - 7.2.14.2. Cross Validation.[./knn/kfold.ipynb and knn folder]
- 7.2.15. Exercise: Apply k-NN on Amazon reviews dataset.

8. Classification algorithms in various situations:

- 8.1. Introduction
- 8.2. Imbalanced vs balanced dataset.
- 8.3. Multi-class classification.
- 8.4. k-NN, given a distance or similarity matrix
- 8.5. Train and test set differences.
- 8.6. Impact of Outliers
- 8.7. Local Outlier Factor.
  - 8.7.1. Simple solution: mean dist to k-NN.
  - 8.7.2. k-distance (A), N(A)
  - 8.7.3. reachability-distance(A, B)
  - 8.7.4. Local-reachability-density(A)
  - 8.7.5. LOF(A)
- 8.8. Impact of Scale & Column standardization.
- 8.9. Interpretability
- 8.10. Feature importance & Forward Feature Selection
- 8.11. Handling categorical and numerical features.
- 8.12. Handling missing values by imputation.
- 8.13. Curse of dimensionality.
- 8.14. Bias-Variance tradeoff.
- 9.14a Intuitive understanding of bias-variance.
- 8.15. Best and worst cases for an algorithm.

9. Performance measurement of models:

- 9.1. Accuracy
- 9.2. Confusion matrix, TPR, FPR, FNR, TNR
- 9.3. Precision & recall, F1-score.
- 9.4. Receiver Operating Characteristic Curve (ROC) curve and AUC.
- 9.5. Log-loss.
- 9.6. R-Squared/ Coefficient of determination.
- 9.7. Median absolute deviation (MAD)
- 9.8. Distribution of errors.

10. Naive Bayes

- 10.1. Conditional probability.
- 10.2. Independent vs Mutually exclusive events.

10.3. Bayes Theorem with examples.
10.4. Exercise problems on Bayes Theorem
10.5. Naive Bayes algorithm.
10.6. Toy example: Train and test stages.
10.7. Naive Bayes on Text data.
10.8. Laplace/Additive Smoothing.
10.9. Log-probabilities for numerical stability.
10.10. Cases:
    10.10.1. Bias and Variance tradeoff.
    10.10.2. Feature importance and interpretability.
    10.10.3. Imbalanced data
    10.10.4. Outliers.
    10.10.5. Missing values.
    10.10.6. Handling Numerical features (Gaussian NB)
    10.10.7. Multiclass classification.
    10.10.8. Similarity or Distance matrix.
    10.10.9. Large dimensionality.
    10.10.10. Best and worst cases.
10.11. Code example
10.12. Exercise: Apply Naive Bayes to Amazon reviews.

11. Logistic Regression:

11.1. Geometric intuition.
11.2. Sigmoid function & Squashing
11.3. Optimization problem.
11.4. Weight vector.
11.5. L2 Regularization: Overfitting and Underfitting.
11.6. L1 regularization and sparsity.
11.7. Probabilistic Interpretation: Gaussian NaiveBayes
11.8. Loss minimization interpretation
11.9. Hyperparameter search: Grid Search and Random Search
11.10. Column Standardization.
11.11. Feature importance and model interpretability.
11.12. Collinearity of features.
11.13. Train & Run time space and time complexity.
11.14. Real world cases.
11.15. Non-linearly separable data & feature engineering.
11.16. Code sample: Logistic regression, GridSearchCV, RandomSearchCV
11.17. Exercise: Apply Logistic regression to Amazon reviews dataset.
11.18. Extensions to Logistic Regression: Generalized linear models (GLM)

12. Linear Regression and Optimization.

12.1. Geometric intuition.
12.2. Mathematical formulation.
12.3. Cases.
12.4. Code sample.
12.5. Solving optimization problems
12.5.1. Differentiation.
13.5.1_a Online differentiation tools
12.5.2. Maxima and Minima
12.5.3. Vector calculus: Grad
12.5.4. Gradient descent: geometric intuition.
12.5.5. Learning rate.
12.5.6. Gradient descent for linear regression.
12.5.7. SGD algorithm
12.5.8. Constrained optimization & PCA
12.5.9. Logistic regression formulation revisited.
12.5.10. Why L1 regularization creates sparsity?
12.5.11. Exercise: Implement SGD for linear regression

13. Support Vector Machines (SVM)
13.1. Geometric intuition.
13.2. Mathematical derivation.
13.3. Loss minimization: Hinge Loss.
13.4. Dual form of SVM formulation.
13.5. Kernel trick.
13.6. Polynomial kernel.
13.7. RBF-Kernel.
13.8. Domain specific Kernels.
13.9. Train and run time complexities.
13.10. nu-SVM: control errors and support vectors.
13.11. SVM Regression.
13.12. Cases.
13.13. Code Sample.
13.14. Exercise: Apply SVM to Amazon reviews dataset.

14. Decision Trees
14.1. Geometric Intuition: Axis parallel hyperplanes.
14.2. Sample Decision tree.
14.3. Building a decision Tree:
14.3.1. Entropy
15.3.1.a Intuition behind entropy
14.3.2. Information Gain
14.3.3. Gini Impurity.

- 14.3.4. Constructing a DT.
- 14.3.5. Splitting numerical features.
- 14.3.5a Feature standardization.
- 14.3.6. Categorical features with many possible values.
- 14.4. Overfitting and Underfitting.
- 14.5. Train and Run time complexity.
- 14.6. Regression using Decision Trees.
- 14.7. Cases
- 14.8. Code Samples.
- 14.9. Exercise: Decision Trees on Amazon reviews dataset.

15. Ensemble Models:

- 15.1. What are ensembles?
- 15.2. Bootstrapped Aggregation (Bagging)
  - 15.2.1. Intuition
  - 15.2.2. Random Forest and their construction.
  - 15.2.3. Bias-Variance tradeoff.
  - 15.2.4. Train and Run-time Complexity.
  - 15.2.5. Code Sample.
  - 15.2.6. Extremely randomized trees.
  - 15.2.7. Cases
- 15.3. Boosting:
  - 15.3.1. Intuition
  - 15.3.2. Residuals, Loss functions and gradients.
  - 15.3.3. Gradient Boosting
  - 15.3.4. Regularization by Shrinkage.
  - 15.3.5. Train and Run time complexity.
  - 15.3.6. XGBoost: Boosting + Randomization
  - 15.3.7. AdaBoost: geometric intuition.
- 15.4. Stacking models.
- 15.5. Cascading classifiers.
- 15.6. Kaggle competitions vs Real world.
- 15.7. Exercise: Apply GBDT and RF to Amazon reviews dataset.

16. Featurizations and Feature engineering.

- 16.1. Introduction.
- 16.2. Time-series data.
  - 16.2.1. Moving window.
  - 16.2.2. Fourier decomposition.
  - 16.2.3. Deep learning features: LSTM
- 16.3. Image data.
  - 16.3.1. Image histogram.
  - 16.3.2. Keypoints: SIFT.

16.3.3. Deep learning features: CNN
16.4. Relational data.
16.5. Graph data.
16.6. Feature Engineering.
16.6.1. Indicator variables.
16.6.2. Feature binning.
16.6.3. Interaction variables.
16.6.4. Mathematical transforms.
16.7. Model specific featurizations.
16.8. Feature orthogonality.
16.9. Domain specific featurizations.
16.10. Feature slicing.
16.11. Kaggle Winners solutions.

17a. **Miscellaneous Topics**
17a.1 Calibration of Models.
17a.1.1 Need for calibration.
17a.1.2 Calibration Plots.
17a.1.3 Platt's Calibration/Scaling.
17a.1.4 Isotonic Regression
17a.1.5 Code Samples
17.a.2 Modeling in the presence of outliers: RANSAC
17.a.3 Productionizing models.
17.a.4 Retraining models periodically.
17.a.5 A/B testing.

17. Unsupervised learning/Clustering: K-Means (2)
17.1. What is Clustering?
18.1.a Unsupervised learning
17.2. Applications.
17.3. Metrics for Clustering.
17.4. K-Means
17.4.1. Geometric intuition, Centroids
17.4.2. Mathematical formulation: Objective function
17.4.3. K-Means Algorithm.
17.4.4. How to initialize: K-Means++
17.4.5. Failure cases/Limitations.
17.4.6. K-Medoids
17.4.7. Determining the right K.
17.4.8. Time and space complexity.
17.4.9. Code Samples
17.4.10. Exercise: Cluster Amazon reviews.

18. Hierarchical clustering

18.1. Agglomerative & Divisive, Dendrograms
18.2. Agglomerative Clustering.
18.3. Proximity methods: Advantages and Limitations.
18.4. Time and Space Complexity.
18.5. Limitations of Hierarchical Clustering.
18.6. Code sample.
18.7. Exercise: Amazon food reviews.

19. DBSCAN (Density based clustering)

19.1. Density based clustering
19.2. MinPts and Eps: Density
19.3. Core, Border and Noise points.
19.4. Density edge and Density connected points.
19.5. DBSCAN Algorithm.
19.6. Hyper Parameters: MinPts and Eps.
19.7. Advantages and Limitations of DBSCAN.
19.8. Time and Space Complexity.
19.9. Code samples.
19.10. Exercise: Amazon Food reviews.

20. Recommender Systems and Matrix Factorization.

20.1. Problem formulation: Movie reviews.
20.2. Content based vs Collaborative Filtering.
20.3. Similarity based Algorithms.
20.4. Matrix Factorization:
20.4.1. PCA, SVD
20.4.2. NMF
20.4.3. MF for Collaborative filtering
20.4.4. MF for feature engineering.
20.4.5. Clustering as MF
20.5. Hyperparameter tuning.
20.6. Matrix Factorization for recommender systems: Netflix Prize Solution [30:00]
20.7. Cold Start problem.
20.8. Word Vectors using MF.
20.9. Eigen-Faces.
20.10. Code example.
20.11. Exercise: Word Vectors using Truncated SVD.

21. Deep Learning:Neural Networks.

21.1. History of Neural networks and Deep Learning.
21.2. How Biological Neurons work?
22.2a Growth of biological neural networks.
21.3. Diagrammatic representation: Logistic Regression and Perceptron
21.4. Multi-Layered Perceptron (MLP).

23.4.3. Softmax Classifier on MNIST dataset.
23.4.4. MLP: Initialization
23.4.5. Model 1: Sigmoid activation.
23.4.6. Model 2: ReLU activation.
23.4.7. Model 3: Batch Normalization.
23.4.8. Model 4 : Dropout.
23.5. MNIST classification in Keras.
23.6. Hyperparameter tuning in Keras.
23.7. Exercise: Try different MLP architectures on MNIST dataset.

24. Deep Learning:Convolutional Neural Nets.
24.1. Biological inspiration: Visual Cortex
24.2. Convolution
24.2.1. Edge Detection on images.
24.2.2. Padding and strides
24.2.3. Convolution over RGB images.
24.3. Convolutional layer.
24.4. Max-pooling.
24.5. CNN Training: Optimization
24.6. Example CNN: LeNet [1998]
24.7. ImageNet dataset
24.8. Data Augmentation.
24.9. Convolution Layers in Keras
24.10. AlexNet
24.11. VGGNet
24.12. Residual Network.
24.13. Inception Network.
24.14. Transfer Learning: Reusing existing models.
24.14.1. What is Transfer learning.
24.14.2. Code example: Cats vs Dogs.
24.15. Code Example: MNIST dataset.
24.16. Assignment: Try various CNN networks on MNIST dataset.

25. Deep Learning:Recurrent Neural Networks
25.1. Why RNNs
25.2. Recurrent Neural Network.
25.3. Training RNNs: Backprop.
25.4. Types of RNNs.
25.5. Need for LSTM/GRU.
25.6. LSTM.
25.7. GRUs.
25.8. Deep RNN.
25.9. Bidirectional RNN.
25.10. Code example : IMDB Sentiment classification

- 25.11. Exercise: Amazon Fine Food reviews LSTM model.

26. Case Study 2: Personalized Cancer Diagnosis.

- 26.1. Business/Real world problem
    - 26.1.1. Overview.
    - 26.1.2. Business objectives and constraints.
- 26.2. ML problem formulation
    - 26.2.1. Data
    - 26.2.2. Mapping real world to ML problem.
    - 26.2.3. Train, CV and Test data construction.
- 26.3. Exploratory Data Analysis
    - 26.3.1. Reading data & preprocessing
    - 26.3.2. Distribution of Class-labels.
    - 26.3.3. “Random” Model.
    - 26.3.4. Univariate Analysis
        - 26.3.4.1. Gene feature.
        - 26.3.4.2. Variation Feature.
        - 26.3.4.3. Text feature.
        - 26.3.4.4. Machine Learning Models
    - 26.3.5. Data preparation.
    - 26.3.6. Baseline Model: Naive Bayes
    - 26.3.7. K-Nearest Neighbors Classification.
    - 26.3.8. Logistic Regression with class balancing
    - 26.3.9. Logistic Regression without class balancing
    - 26.3.10. Linear-SVM.
    - 26.3.11. Random-Forest with one-hot encoded features
    - 26.3.12. Random-Forest with response-coded features
    - 26.3.13. Stacking Classifier
    - 26.3.14. Majority Voting classifier.
- 26.4. Assignments.

27. Case study 3:Taxi demand prediction in New York City.

- 27.1. Business/Real world problem.
    - 27.1.1. Overview
    - 27.1.2. Objectives and Constraints
- 27.2. Mapping to ML problem
    - 27.2.1. Data
    - 27.2.2. dask dataframes
    - 27.2.3. Fields/Features.
    - 27.2.4. Time series forecasting/Regression.
    - 27.2.5. Performance metrics.
- 27.3. Data Cleaning
    - 27.3.1. Latitude and Longitude data

27.3.2. Trip Duration.
27.3.3. Speed.
27.3.4. Distance.
27.3.5. Fare.
27.3.6. Remove all outliers/erroneous points.
27.4. Data Preparation
27.4.1. Clustering/Segmentation
27.4.2. Time binning
27.4.3. Smoothing time-series data.
27.4.4. Time series and Fourier transforms.
27.5. Baseline models
27.5.1. Ratios and previous-time-bin values.
27.5.2. Simple moving average.
27.5.3. Weighted Moving average.
27.5.4. Exponential weighted moving average.
27.5.5. Results.
27.6. Regression models:
27.6.1. Train-Test split & Features
27.6.2. Linear regression.
27.6.3. Random Forest regression.
27.6.4. Xgboost Regression.
27.6.5. Model comparison.
27.7. Assignment.

28. Case Study 4: Microsoft Malware Detection
28.1. Business/real world problem
28.1.1. Problem definition
28.1.2. Objectives and constraints.
28.2. Machine Learning problem mapping
28.2.1. Data overview.
28.2.2. ML problem.
28.2.3. Train and test splitting.
28.3. Exploratory Data Analysis
28.3.1. Class distribution.
28.3.2. Feature extraction from byte files
28.3.3. Multivariate analysis of features from byte files.
28.3.4. Train-Test class distributions
28.4. ML models - using byte files only
28.4.1. Random Model.
28.4.2. k-NN
28.4.3. Logistic regression
28.4.4. Random Forest & Xgboost
28.5. ASM Files

28.5.1. Feature extraction & Multi-threading.
28.5.2. File-size feature
28.5.3. Univariate analysis on ASM features.
28.5.4. t-SNE analysis
28.5.5. ML models on ASM file features
28.6. Models on all features
28.6.1. t-SNE
28.6.2. RandomForest and Xgboost
28.7. Assignments.

29. Case study 5: Netflix Prize

29.1. Business/Real world problem
29.1.1. Problem definition
29.1.2. Objectives and constraints.
29.2. Mapping to an ML problem
29.2.1. Data overview.
29.2.2. ML problem formulation.
29.3. Exploratory Data Analysis
29.3.1. Data preprocessing
29.3.2. Temporal Train-Test split.
29.3.3. Preliminary data analysis.
29.3.4. Sparse matrix representation.
29.3.5. Average ratings for various slices.
29.3.6. Cold start problem.
29.3.7. Computing Similarity matrices
29.3.8. User-User similarity matrix
29.3.9. Movie-Movie similarity.
29.3.10. Does movie-movie similarity work?
29.4. ML Models
29.4.1. SurPRISE library
29.4.2. Overview of the modelling strategy.
29.4.3. Data Sampling.
29.4.4. Google drive with intermediate files
29.4.5. Featurizations for regression.
29.4.6. Data transformation for SurPRISE.
29.4.7. Xgboost with 13 features
29.4.8. Surprise Baseline model.
29.4.9. Xgboost + 13 features +Surprise baseline model
29.4.10. Surprise KNN predictors
29.4.11. Matrix Factorization models using Surprise
29.4.12. SVD ++ with implicit feedback
29.4.13. Final models with all features and predictors.
29.4.14. Comparison between various models.

29.4.15. Assignments

30. Case study 6:StackOverflow.

30.1. Business/Real world problem

30.1.1. Problem description.

30.1.2. Business objectives and constraints.

30.2. Mapping to an ML problem

30.2.1. Data overview

30.2.2. ML problem formulation.

30.2.3. Performance metrics.

30.2.4. Hamming loss

30.3. EDA

30.3.1. Data Loading.

30.3.2. Analysis of tags

30.3.3. Data preprocessing.

30.4. ML modeling

30.4.1. Multi-label classification.

30.4.2. Data preparation.

30.4.3. Train-test split.

30.4.4. Featurization.

30.4.5. Logistic regression: One VS Rest

30.4.6. Sampling data and tags + Weighted models.

30.4.7. Logistic regression revisited.

30.4.8. Why not use advanced techniques?

30.5. Assignments.

31. Case study 7:Quora Question pair similarity

31.1. Business/Real world problem.

31.1.1. Problem definition.

31.1.2. Business objectives and constraints.

31.2. Mapping to an ML problem

31.2.1. Data overview

31.2.2. ML problem and performance metric.

31.2.3. Train-test split

31.3. EDA

31.3.1. Basic Statistics.

31.3.2. Basic Feature Extraction.

31.3.3. Text Preprocessing.

31.3.4. Advanced Feature Extraction.

31.3.5. Feature analysis.

31.3.6. Data Visualization: T-SNE.

31.3.7. TF-IDF weighted word-vector featurization.

31.4. ML Models

31.4.1. Loading data.
31.4.2. Random Model.
31.4.3. Logistic Regression & Linear SVM
31.4.4. XGBoost
31.5. Assignments.

32. Case study 8:Self-Driving Car

32.1. Problem definition.
32.2. Datasets.
32.3. Data understanding & Analysis.
32.3.1. Files and folders.
32.3.2. Dash-cam images and steering angles.
32.3.3. Split the dataset: Train vs Test
32.3.4. EDA: Steering angles
32.3.5. Mean Baseline model: simple .

32.4. Deep-learning model
32.4.1. Deep Learning for regression: CNN, CNN+RNN
32.4.2. Batch load the dataset.
32.4.3. NVIDIA's end to end CNN model.
32.4.4. Train the model.
32.4.5. Test and visualize the output.
32.4.6. Extensions.
32.5. Assignment.

33. Case studies/Projects:

33.1. Amazon Fine Food Reviews
33.2. Personalized Cancer Diagnosis.
33.3. Taxi demand prediction in New York City.
33.4. Microsoft Malware Detection
33.5. Netflix Movie Recommendation
33.6. StackOverflow Tag Prediction
33.7. Quora Question pair similarity
33.8. Self-Driving Car
33.9. Amazon fashion discovery engine.
33.10. Song Similarity engine.
33.11. Human Activity Recognition using mobile phone's accelerometer and gyroscope data.
33.12. Which ad to show to which user: Ad Click prediction.
33.13. Music Generation.
33.14. Airbnb First Travel Destination.

33.15. Facebook Friend Recommendation System