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
        1.4.5. pass statement
    1.5. Functions
        1.5.1. Introduction
        1.5.2. Types of functions
        1.5.3. Function Arguments
        1.5.4. Recursive Functions
        1.5.5. Lambda Functions
        1.5.6. Modules
        1.5.7. Packages
    1.6. Object Oriented Programming
        1.6.1. Class
        1.6.2.
    1.7. File Handling
    1.8. Exception Handling

        1.8.1. Variables
        1.8.2. Statements
        1.8.3. Conditional Statements
        1.8.4. Loops
        1.8.5. Functions
        1.8.6. Data types (lists, dict, sets, tuple, string)
        1.8.7. Classes and packages.
        1.8.8. File Handling
        1.8.9. Exception Handling
    2.3. Pandas
        2.3.1 Getting started with pandas

2.3.2 Data Frame Basics
2.3.3 Loading Data from csv, excel, txt.. etc
2.3.4 Handling Missing data
2.3.5 Group by
2.3.6 Concat and merging data
2.3.7 Pivot Table
2.3.8 Reshaping of Table
2.3.9 Time series

1.9. Matplotlib
1.10. Numpy and Scipy
1.11. Seaborn
1.12. Scikit Learn
1.13. Debugging Python
1.14. Computational Complexity: an Introduction
  1.14.1. Space and Time Complexity.
  1.14.2. Examples:
    1.14.2.1. Multiply a matrix with a vector.
    1.14.2.2.
  1.14.3. Further reading.

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.

3.2. 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 distributions.
- 3.2.8. Skewness
- 3.2.9. Kurtosis
- 3.2.10. Standard normal variate (z) and standardization.
- 3.2.11. Kernel density estimation.
- 3.2.12. Central Limit theorem with examples.
- 3.2.13. Real world problems:
  - 3.2.13.1. Estimate mean and variance robustly in the presence of noise/outliers.
  - 3.2.13.2. How to test if a random variable is normally distributed or not? Q-Q plot.

3.3. Uniform Distribution and random number generators
- 3.3.1. Parameters of a distribution.
- 3.3.2. PDF
- 3.3.3. CDF
- 3.3.4. Random number generators.

3.4. Log-normal and power law distribution:
- 3.4.1. Examples: incomes,
- 3.4.2. CDF, PDF
- 3.4.3. Converting a log-normal distribution to normal.
- 3.4.4. Haberman dataset features.
- 3.4.5. PDF of power-law distributions.
- 3.4.6. Converting power law distributions to normal: Box-Cox transform.

3.5. Correlation
- 3.5.1. Co-variance
- 3.5.2. Pearson Correlation Coefficient
- 3.5.3. Spearman Rank Correlation Coefficient
- 3.5.4. Correlation vs Causation

3.6. Confidence Intervals
- 3.6.1. For mean of a normal random variable
  - 3.6.1.1. Known std-deviation
  - 3.6.1.2. Unknown std-deviation (using t-distribution)
- 3.6.2. Distribution dependent C.I using simulations.

3.6.3. Using bootstrapping.
3.7. Hypothesis testing
3.7.1. Why learn Hypothesis testing?
3.7.2. Testing methodology, Null-hypothesis, test-statistic, p-value.
3.7.3. Resampling and permutation test.
3.7.4. K-S Test for similarity of two distributions.
3.7.5. Example from Iris flower dataset.

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. Data Matrix
5.1.1. Represent a dataset as a Matrix.
5.1.2. Normalization, Standardization, Centering and Scaling.
5.1.3. Mean, Variance, Co-variance of a Data Matrix.
5.1.4. Symmetric matrix.
5.2. MNIST dataset (784 dimensional)
5.2.1. Explanation of the dataset.
5.2.2. Code to load this dataset.
5.3. Principal Component Analysis.
5.3.1. Why learn it.
5.3.2. Geometric intuition.
5.3.3. Mathematical objective function.
5.3.4. Eigenvalues and eigenvectors.
5.3.5. PCA for dimensionality reduction and visualization.
5.3.6. Visualize MNIST dataset.
5.3.7. Limitations of PCA.
5.3.8. Code example.
5.4. T-distributed stochastic neighborhood embedding (t-SNE)
5.4.1. Neighborhood of a point.
5.4.2. Neighborhood embedding.
5.4.3. Geometric intuition.
5.4.4. Mathematical formulation
5.4.4.1. Define distance between points using an exponential function.
5.4.4.2. Why t-disb?

- 5.4.4.3. Optimization problem.
- 5.4.5. How to apply t-SNE and interpret its output (distill.pub)
- 5.4.6. t-SNE on MNIST.

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

- 6.1. Basics:
  - 6.1.1. Amazon product reviews overview.
  - 6.1.2. Sentiment polarity: Positive and Negative.
  - 6.1.3. Dataset deep-dive.
  - 6.1.4. iPython code for analysis.
- 6.2. Featurizations: convert text to numeric vectors.
  - 6.2.1. Bag of words.
  - 6.2.2. Preprocessing: Stemming, Stopping, Lemmatization,
  - 6.2.3. Tf-idf (term frequency- inverse document frequency)
  - 6.2.4. Word2Vec, Avg-Word2Vec, tf-idf weighted Word2Vec.
  - 6.2.5. Code samples.
- 6.3. Exercise: t-SNE visualization of Amazon reviews

7. Classification and Regression Models.

- 7.1. Foundations
  - 7.1.1. Classification vs Regression (examples)
  - 7.1.2. Data matrix notation.
  - 7.1.3. Decision surface.
- 7.2. K-Nearest Neighbors
  - 7.2.1. Geometric intuition with a toy example.
  - 7.2.2. Smoothness assumptions.
  - 7.2.3. Distance measures: Euclidean, Manhattan, Hamming
  - 7.2.4. Simple implementation:
    - 7.2.4.1. Majority vote.
    - 7.2.4.2. Pseudo code.
    - 7.2.4.3. Train time and space complexity
    - 7.2.4.4. Test time and space complexity.
    - 7.2.4.5. Limitations.
  - 7.2.5. Determining the right “k”
    - 7.2.5.1. Cross validation.
    - 7.2.5.2. K-fold cross validation.
    - 7.2.5.3. Train, Test and Cross validation.
    - 7.2.5.4. Overfitting and Underfitting.
  - 7.2.6. k-NN for regression.
  - 7.2.7. Decision surface and voronoi tessellation.
  - 7.2.8. kd-tree based k-NN:
    - 7.2.8.1. kd-tree geometric intuition.
    - 7.2.8.2. How to build a kd-tree.
    - 7.2.8.3. Time and Space complexity.

- 7.2.8.4. Limitations.
- 7.2.9. Locality sensitive Hashing (LSH)
  - 7.2.9.1. Geometric intuition.
  - 7.2.9.2. Hashing functions and distance measures.
- 7.2.10. Code Samples:
- 7.2.11. References and further reading.
- 7.2.12. Exercise: Apply k-NN on Amazon reviews dataset.

7.3. Performance measurement of models:
- 7.3.1. Accuracy
- 7.3.2. Confusion matrix, TPR, FPR, FNR, TNR
- 7.3.3. Precision and recall.
- 7.3.4. Receiver Operating Characteristic Curve (ROC) curve and AUC.
- 7.3.5. Log-loss.
- 7.3.6. R-Squared.
- 7.3.7. Median absolute deviation (MAD)

7.4. Naive Bayes
- 7.4.1. Conditional probability.
- 7.4.2. Conditional independence.
- 7.4.3. Bayes rule and examples.
- 7.4.4. Naive Bayes algorithm.
- 7.4.5. Toy example.
- 7.4.6. Space and Time complexity: train and test time.
- 7.4.7. Laplace/Additive Smoothing
- 7.4.8. Underfitting and Overfitting.
- 7.4.9. Feature importance and interpretability.
- 7.4.10. Exercise: Apply Naive Bayes to Amazon reviews.

7.5. Logistic Regression:
- 7.5.1. Geometric intuition.
- 7.5.2. Sigmoid function: Squashing
- 7.5.3. Mathematical formulation of Objective function.
- 7.5.4. Weight vector.
- 7.5.5. Regularization: Overfitting and Underfitting.
- 7.5.6. L2 regularization.
- 7.5.7. L1 regularization and sparsity.
- 7.5.8. Probabilistic Interpretation: Gaussian Naive Bayes.
- 7.5.9. Loss function interpretation:
  - 7.5.9.1. 0-1 loss.
  - 7.5.9.2. Log-loss.
  - 7.5.9.3. Other loss functions: Hinge, Squared loss.
- 7.5.10. Centering and Scaling of columns.
- 7.5.11. Feature importance and interpretability.
- 7.5.12. Collinearity of features.
  - 7.5.12.1. Definition.

7.5.12.2. Determining Collinearity.
7.5.12.3. Removing collinear features.
7.5.13. Featurizing categorical features: one-hot encoding.
7.5.14. Featuring Nominal features.
7.5.15. Test/Run time space and time complexity.
7.5.16. Internet scale: Large data and low-latency.
7.5.17. Decision surface and examples.
7.5.18. Exercise: Apply Logistic regression to Amazon reviews dataset.

7.6. Linear Regression:
7.6.1. Geometric intuition.
7.6.2. Mathematical formulation.
7.6.3. Squared loss and loss-function based interpretation.
7.6.4. toy-example.

7.7. Solving optimization problems : Stochastic Gradient Descent.
7.7.1. Gradient, derivative, slope, partial derivative.
7.7.2. Gradient descent: geometric intuition.
7.7.3. Rate of convergence.
7.7.4. SGD: algorithm and rate of convergence.
7.7.5. Constrained optimization and Penalty method.
7.7.6. Exercise: Implement SGD for linear regression.

7.8. Bias-Variance tradeoff
7.8.1. Intuition: Underfit and Overfit.
7.8.2. Derivation for linear regression.
7.8.3. Bias Variance tradeoff for k-NN, NaiveBayes, Logistic Regression, Linear regression.

7.9. Support Vector Machines (SVM)
7.9.1. Geometric intuition.
7.9.2. Mathematical derivation.
7.9.3. Loss function (Hinge Loss) based interpretation.
7.9.4. Support vectors.
7.9.5. Linear SVM.
7.9.6. Primal and Dual.
7.9.7. Kernelization.
7.9.8. RBF-Kernel.
7.9.9. Polynomial kernel.
7.9.10. Domain specific Kernels.
7.9.11. Train and run time complexities.
7.9.12. Bias-variance tradeoff: Underfitting and Overfitting
7.9.13. nu-SVM: control errors and support vectors.
7.9.14. SVM Regression.
7.9.15. Code Samples.
7.9.16. Exercise: Apply SVM to Amazon reviews dataset.

7.10. Decision Trees

- 7.10.1. Geometric Intuition: Axis parallel hyperplanes.
- 7.10.2. Nested if-else conditions.
- 7.10.3. Sample Decision tree.
- 7.10.4. Building a decision Tree:
  - 7.10.4.1. Entropy, Information Gain
  - 7.10.4.2. Gini Impurity (CART)
  - 7.10.4.3. Depth of a tree: Geometric and programming intuition.
  - 7.10.4.4. Categorical features with many levels.
- 7.10.5. Regression using Decision Trees.
- 7.10.6. Bias-Variance tradeoff.
- 7.10.7. Limitations
- 7.10.8. Code Samples.

7.11. Ensemble Models:
- 7.11.1. Bootstrapped Aggregation (Bagging)
  - 7.11.1.1. Random Forest and their construction.
  - 7.11.1.2. Bias-Variance tradeoff
  - 7.11.1.3. Applicative details.
  - 7.11.1.4. Code Samples.
- 7.11.2. Boosting:
  - 7.11.2.1. Intuition
  - 7.11.2.2. Gradient Boosting and XGBoost
    - 7.11.2.2.1. Algorithm.
    - 7.11.2.2.2. Loss function and advantages.
    - 7.11.2.2.3. XGBoost code samples.
  - 7.11.2.3. AdaBoost: geometric intuition.
- 7.11.3. Cascading models
- 7.11.4. Stacking models.
- 7.11.5. How to win Kaggle competitions using Ensembles.
- 7.11.6. Exercise: Apply GBDT and RF to Amazon reviews dataset.

8. Unsupervised learning: Clustering

8.1. K-Means
- 8.1.1. Geometric intuition, Centroids
- 8.1.2. Mathematical formulation: Objective function
- 8.1.3. K-Means Algoithm.
- 8.1.4. How to initialize: K-Means++
- 8.1.5. Failure cases/Limitations.
- 8.1.6. K-Medoids
- 8.1.7. Kernel K-Means and Spectral Clustering
- 8.1.8. Determining the right K.
- 8.1.9. Time and space complexity.
- 8.1.10. Code Samples
- 8.1.11. Exercise: Cluster Amazon reviews.

8.2. Hierarchical clustering

8.2.1. Agglomerative vs Divisive.
8.2.2. Agglomerative Algorithm.
8.2.3. MIN, MAX, Average methods.
8.2.4. Advantages.
8.2.5. Limitations.
8.2.6.

8.3. DBSCAN (Density based clustering)
8.3.1. MinPts and Eps: Density
8.3.2. Core, Border and Noise points.
8.3.3. Density edge and Density connected points.
8.3.4. Algorithm.
8.3.5. Determining the optimal Hyper Parameters: MinPts and Eps.
8.3.6. Sensitivity issues of DBSCAN.

9. Recommender Systems and Matrix Factorization.

9.1. Problem formulation: IMDB Movie reviews.
9.2. Content based vs Collaborative Filtering.
9.3. Item-Item and User-User Similarity based Algorithms.
9.4. Matrix Factorization:
9.4.1. PCA, SVD
9.4.2. MF
9.4.3. NMF
9.4.4. NMF vs Clustering.
9.5. Matrix Factorization for recommender systems: Netflix Prize Solution
9.5.1. Mathematical Optimization problem.
9.5.2. Item and User biases.
9.5.3. Time varying ratings.
9.6. Matrix Factorization for feature engineering: word2Vec

10. Miscellaneous topics

10.1. Handling missing values.
10.2. Modeling in the presence of outliers: RANSAC
10.3. Productionizing models:
10.3.1. Real world constraints: speed, interpretability, size
10.3.2. Retraining models periodically as needed
10.4. A/B testing

11. Neural Networks and Deep Learning:

11.1. Classical Neural Nets.
11.1.1. Diagrammatic representation, Activation functions.
11.1.2. Mathematical formulation.
11.1.3. Back propagation and chain rule of differentiation.
11.1.4. Vanishing Gradient problem.
11.1.5. Bias-Variance tradeoff.
11.1.6. Determining the number of levels.
11.1.7. Decision surfaces.

    11.1.8. Code Samples.
  11.2. AutoEncoder
  11.3. Modern activation functions.
  11.4. Convolutional Neural Nets.
  11.5. Long Short-term memory (LSTMs)
  11.6. Transfer learning: reusing pre trained models.

12. Case studies/Projects:

  12.1. Amazon fashion discovery engine.
  12.2. Malware Detection on Windows OS.
  12.3. Song Similarity engine.
  12.4. Predict customer propensity to purchase using CRM data.
  12.5. Suggest me a movie to watch: Netflix Prize.
  12.6. Human Activity Recognition using mobile phone's accelerometer and gyroscope data.
  12.7. Which ad to show to which user: Ad Click prediction.