Python Basis

List Slicing list[start:stop:step] [start, stop)

NumPy & Pandas

Memory Int32 blocks is 4 bytes, Int64 blocks is 8 bytes. Axis 0 is row, Axis 1 is column.
Joins Inner Join: intersection of keys, Outer Join: union of keys, Left Join: all keys from left table, Right Join: all keys from right table.
GroupBy split-apply-combine. Split the data into groups based on some criteria. Apply a function to each group independently. Combine the results into a data structure. ex: df.groupby('column_name').mean()
Lambda anonymous function. ex: df.apply(lambda x: x + 1)
Aggregate perform multiple operations on grouped data. ex: df.groupby('column_name').agg('col1': 'mean', 'col2': 'sum')

Text Data

Text Cleaning lower(), upper(), strip(): (remove whitespace), split(a): (split string into list at a), replace(a,b): (replace a with b).

Regex

Regex ^: start of string, $: end of string, . : any character except newline, * : 0 or more, + : 1 or more, ? : 0 or 1, \s : whitespace, \S : non-whitespace, \d : digit, \D : non-digit, \w : word character (alphanumeric + _), \W : non-word character.
RE functions findall(): returns all non-overlapping matches of pattern in string, search(): searches for pattern in string and returns a match object, sub(): replaces occurrences of pattern with repl in string. split(): splits string by occurrences of pattern.
Grouping Regex parentheses () to create groups. ex: (\d3)-(\d2)-(\d4) matches a pattern like 123-45-6789 and creates three groups: 123, 45, and 6789. brackets [] to create character classes. ex: [aeiou] matches any vowel.

NLP & Sentiment Analysis

Sentiment Analysis A technique used to determine the sentiment or emotion expressed in a piece of text. It can be used to analyze customer reviews, social media posts, and other forms of text data to gain insights into customer opinions and feelings.
Vader Valence Aware Dictionary and sEntiment Reasoner. Short emotive sentiment lexicon tool for text, social media. Encodes slang, emojis, acronyms and punctuation/case.
Bag of Words Document as a vector of word counts, ignoring grammar and word order but keeping multiplicity.
TF Term Frequency: $tf_{i,j} = \frac{n_{i,j}}{\sum_k n_{k,j}}$ where $n_{i,j}$ is the number of times term t_i appears in document d_j and the denominator is the total number of terms in document d_j. Can also use binary (1 if present, 0 if not) or log scalings.
IDF Inverse Document Frequency: $idf_i = log\frac{|D|}{|\{j:t_i \in d_j\}|}$ where $|D|$ is the total number of documents and the denominator is the number of documents containing term t_i. Defined for words, not documents. Want to show how important a word is to a document. Higher IDF means more important. Lower IDF means more common.
TF-IDF $tfidf_{i,j} = tf_{i,j} * idf_i$ Scaling the term frequency by the inverse document frequency. Helps to reduce the weight of common words and increase the weight of rare words.
Cosine Similarity measures the cosine of the angle between two non-zero vectors. How similar are two documents.
Manhattan Distance sum of absolute differences between two vectors. $|x - y|_1 = \sum |x_i - y_i|$
WordNet A lexical database for the English language. Groups words into sets of synonyms called synsets, provides short definitions and usage examples, and records various semantic relations between these synonym sets.

Language Models

Bag of words Surprisingly good performance on text classification tasks, but ignores word order and context.
Probabilistic LM $P(w_i|w_1, w_2, ..., w_{i-1})$ Probability of word given previous words. Approx: $P(w_i|w_{i-n+1}, ..., w_{i-1}) = \frac{\#w_{i-n+1,...}w_i}{\#w_{i-n+1,...}w_{i-1}}$
Word2Vec Represents words as dense vector embeddings. Vectors capture semantic relationships between words. CBOW: predicts a word given its context. Skip-gram: predicts context words given a target word.

Data Visualization

NORI Nominal: data can be cat (Mode), Ordinal: Categorical and ranked (Median), Ratio: Cat ranked evenly spaced(mean), Interval: cat, ranked, and even spaced and natural zero (anything).
1D: Bar chart (cat: NO), Stacked bar, Pie chart (cat: NO), Histogram (num: RI)
2D: Scatter plot (num, num: RI), Line plot (x: time (RI), y: num (RI)), Box plot (x: cat (NO), y: num (RI)), Heatmap (cat or num, cat or num: all)
3D+: 3D scatter plot (num, num, num: RI), Bubble chart (num, num, num: RI), Pair plot (num, num, num: all)

Matrix and Linear Algebra

Word2Vec Semantic meaning of words based on embeddings
Vector Space $v+u \in V$, $cv \in V$, $v+u = u+v$, $(u+v)+w = u+(v+w)$, $v+0 = v$, $v+(-v) = 0$, $c(u+v) = cu+cv$, $(c+d)v = cv+dv$, $c(dv) = (cd)v$, $1v = v$
Cosine: $cos(\theta) = \frac{u \cdot v}{||u||||v||}$
Data Redundancy Cosine Sim, Jaccard Sim (for sets), Hamming Dist (for binary features). Hash based uniqueness check, Statistical entropy: $H(X) = -\sum p(x)log(p(x))$
Sparce Matrix COO: store row, col, data arrays. CSR: store row pointers, col indices, data arrays. CSC: store col pointers, row indices, data arrays.
SVD Method of simplifying complex data. $A = U\Sigma V^T$. U: left singular vectors (eigenvectors of AA^T) $m \times m$. Σ: singular values (square roots of eigenvalues of A^TA) $m \times n$. V: right singular vectors (eigenvectors of A^TA) $n \times n$.
PCA Goal to find a lower-dimensional representation of the data that captures as much of the variance as possible. Center the data, compute covariance matrix, compute eigenvalues and eigenvectors, sort eigenvectors by eigenvalues, select top k eigenvectors, project data onto new subspace.

Probability

Bayes Theorem $P(A|B) = \frac{P(B|A)P(A)}{P(B)} = P(A \cap B)/P(B)$
Pearson Correlation coefficient $r = \frac{cov(X,Y)}{\sigma_X \sigma_Y} = \frac{\sum_i (X_i - \bar{X})(Y_i - \bar{Y})}{\sqrt{\sum_i (X_i - \bar{X})^2}\sqrt{\sum_i (Y_i - \bar{Y})^2}}$
R^2 the ratio of the variance explained by the model to the total variance.
Margin Of Error the range of uncertain accociated with a sample statistic.
Linear Regression $y = \beta_0 + \beta_1 x$ where $\beta_1 = \frac{cov(X,Y)}{var(X)} = \frac{\sum (X-\bar{X})(Y-\bar{Y})}{\sum (X-\bar{X})^2}$ and $\beta_0 = \bar{Y} - \beta_1 \bar{X}$
Linear Regression Worked Example $(x,y) = \{(1,2),(2,3),(3,5)\}$ $\bar{X} = 2$, $\bar{Y} = 10/3$. $X - \bar{X} = \{-1,0,1\}$, $Y - \bar{Y} = \{-4/3, -1/3, 5/3\}$. $cov(X,Y) = \frac{(-1)(-4/3)+(0)(-1/3)+(1)(5/3)}{3} = 3/3 = 1$. $var(X) = \frac{(-1)^2+0^2+1^2}{3} = 2/3$. $\beta_1 = \frac{cov(X,Y)}{var(X)} = \frac{1}{2/3} = 3/2$. $\beta_0 = \bar{Y} - \beta_1 \bar{X} = 10/3 - (3/2)(2) = 10/3 - 3 = 1/3$. $y = 1/3 + (3/2)x$.

IMPT Pandas functions

fillna(): fills NA/NaN values using the specified method.
dropna(): removes missing values.
value_counts(): returns a Series containing counts of unique values.
loc[]: access a group of rows and columns by labels or a boolean array.
iloc[]: access a group of rows and columns by integer position.