|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Fraud | | 1) Profit motive. 2) Deception of identity. 3) Organised collusion  Types: Incentive abuse, insurance fraud, scams, invoice fraud | | | - Knowlege factor (something you know): password, security qn, pin  - Posession factor (something you have): phone, card, hardware token  - Inherence factor (something you are): fingerprint, retina, face recognition | | | | |
| Fraud Detection Systems | | 1. Expert-based approach: based on intuition, experience, domain knowledge. Time consuming, manual investigations  2. Heuristic approach: rule-engines built of if-else patterns. Easy to evade once rules are known, expensive to built  3. Data-drive approach: risk score models, anomaly detection, higher precision over big datasets. Improved operational efficiency | | | | | | | |
| Data | | RFM: recency (how much time since last trx); freq; monetary (average, mean, median value of trx)  Graph data. Can capture hidden attributes btw users. Feature scaling for gradient descent or distance-based algos  Normalisation (min-max scaling): rescales values to range btw 0 and 1. Standardisation: centers data around mean 0, s.d. = 1 | | | | | | | |
| Model | | Metrics: precision if false postivies/banning wrong legitimate users more costly. Recall if false negative, or fraud not detected more costly  Interpretability. Standards for operational efficiency. Economical costs vs benefits. Regulatory compliance (GDPR, PDPR) | | | | | | | |
| Network analysis | | Konigsberg Bridges. Eulerian Path: Traverse all edges exactly once w/o repeat. Euler's soln: all nodes in graph must be connect & graph must have exactly 0 or 2 nodes of odd degee  Eulerian circuit: cross all edge and start and end at same place. All eulerian circuits are eulerian pahts but not the other way.  If path/circuit exists, can find using Fleury's Algo  Traveling Salesman problem. Shortest route that visits each city exactly once, and returns to origin city = Hamiltonian path | | | | | | | |
| Types of Network analysis | | 1. Topological: overall structure of network (num and types of nodes and connections btw them)  2. Influence: actions of 1 or more nodes can influence other nodes  3. Flow: flow of info through a network  4. Dynamics: how network change over time, including formation and dissolution of connections btw nodes | | | | 5. Resilience: ability of network to withstand disruptions or failues, and how it can recover  6. Predictive: make predictions abt future events  7. Centrality: how central/impt a node is within a network  8. Community detection: identify groups within a network (modularity optimization, DBSCAN, spectral clustering) | | | |
| Graphs | G = (V,E). V = {v1, v2, …, vn}. .  N(a) = nodes connected to vertex a  Weighted graph: G = (V, E, w). w = mapping from set of edges to set of numbers  Types of weighted graph: 1) Binary (1 and 0)  2) Signed (-1, 0, 1). 3) Numeric.  4) Normalised (all outgoing edges of node sum to 1)  5) Jaccard weight = intersection/union | | | | | | | Degree = deg(v) = num of edges connecting to that node  0 deg = isolated. 1 degree = endvertex/leaf. In-degree/out-degree  Degree of graph =  Density =  Graph is dense if density close to 1 or sparse if close to 0 | |
| Bipartite Graph | Bipartite graph: G = (U, V, E) where U, V are 2 disjoint sets of vertices  . Usually undirected edges | | | Pros: Clarity of data vis in network structures. Scalabilities (limited num of edges)  Suited for centrality calculations within a set, rather than a node or whole network  Suited for recommendation problems, via collaborative filtering for similar users | | | | | |
| Directed graph | | Directed graph: G = (V, E). But now if (p, q) E, then p = source and q = sink of edge  In-degree = deg-in(v) = num of nodes pointing toward target/sink node  Out-degree = deg-out(v) = num of nodes that can be reached from source node | | | | | | | Degree of vertex = deg(v) = deg-in(v) + deg-out(v)  Source: deg-in(v) = 0 and deg-out(v) > 0  Sink: deg-in(v) = 0 and deg-out(v) = 0 |
| Edges | | 1) Edge list. E.g. [(1,2), (2,5), …]  O(E) access time = num of edges to search to verify if edge exists  2) Adjacency Matrix. . aij = 1 if there is a edge from i to j  3) Adjacency List. {1: [[2, 1], [3, 1]], 2: [[3, 3]]} | | | | | Undirected graph w n nodes, max edges = E =  Directed graph, max edges = E = n(n-1)  Adjacency matrix is symmetrical for undirected graph  Adj matrix2 = num of distinct paths btw nodes present, if value != 0  # source\_node: [[sink\_node, weight], [sink\_node, weight], …] | | |
|  | | |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | | Op | Memory | Add vertex | Add edge | Remove vertex | Remove edge | Query for edge (x, y) | Query iteration over edge (x, y) | | Adj matrix | O(V2) | O(V2) | O(1) | O(V2) | O(1) | O(1) | O(V2) | | Adj List | O(V+E) OR O(max(V, E)) | O(1) | O(1) | O(V+E) | O(E) | O(V) OR O(min(V, E)) | O(V+E) |   In general, use adj list for sparse graph | | | | | | | |
| Network Qualities | | Dyadicity  Homophily: people have a strong tendency to associate with others whom they perceive as similar to themselves (Opposite: Heterophily)  Network has homophilic qualities if nodes of a classificiation are significantly more connected to other nodes of the same classification, than to non-similar nodes.  Let network G = (V, Eg) where each vertice is assigned fraud w prob p, and non-fraud w prob 1-p  Let G' = (V, E) be a random sample of G w p fraction of fraud, and 1 - p fraction of non-fraud vertices  Cross-edges = edges that connect vertices w differently labelled attributes  Consider any edge (i, j) Eg. Let random var Xij = 1 if it is a cross-edge and Xij = 0 otherwise  Then Xij is a Bernoulli r.v. s.t. P(Xij = 1) = 2p(1-p). = E(Xij) = 2p(1-p). = 2p(1-p)(1 - 2p(1-p)) | | | | | | | |
| Detection | | Test for homophily. Let be the fraction of cross-labelled edges in G. H0: ≥ against H1: < (observed < expected).  H0 = network is not homophilic. = expected value = 2p(1-p). 1-tailed t-test only acceptable for small network  T-test assumption of edges being independent is violated for networks in general use non-parametric bootstrapping over multiple samples | | | | | | | |
| Measures of centrality | | | Centrality = what characterises an impt node.An impt node is one which many paths flow through this node OR how cohesive the node is in the network connectivity, in relation to degrees. Shortest path has shortest num of edges | | | | | | |
| 1) Betweeness: Nodes/edges that more frequently lie on the shortest path btw other nodes/edges will have higher betweeness centrality  Betweeness centrality of a vertex = , where = total num of shortest path from vertex s to t and = num of shortest paths from vertex s to t, that also pass through vertex v. Sum over all possible vertice pairs of s and t, excluding the node v  Betweenness centrality of a network =  Brandes' algo can help compute shortest paths in graph. Scores must be normalized by num of nodes | | | | | | |
| 2) Degree = num of edges a node has. n = num of nodes in graph  Undirected graph: . Directed graph:  Degree centrality of network = | | | | | | |
| 3) Closeness = average length of the shortest path btw a node and all other nodes in the graph . (Opposite: Farness)  , where d(v,u) = shortest dist btw node u and all other nodes v. Normalise by multiplying by n-1  Closeness centrality of network = | | | | | | |
| 4) Eigenvector centrality = measure of influence of a given node in a network (Google's PageRank, Katz Centrality)  Nodes w high eigenvector centrality will be connected to other nodes w high eigenvector centrality | | | | | | |
| Matrices for Bipartite graph | | |  |  |  |  | | --- | --- | --- | --- | |  | v1 | v2 | v3 | | u1 | 1 | 1 | 0 | | u2 | 0 | 1 | 1 |   Use reduced adjacency matrix. Represent the 2 set of nodes as rows and columns in matrix  . ATA = Group-Group Affliation =  AAT = Person-Person Affliation = | | | | | | | |
| Betweenness centrality for vertex u of bipartite graph = | | | | | | | |
| Degree for vertex u in set U: . Degree for vertex v in set V: : | | | | | | | |
| Closeness centrality for vertex u of bipartite graph = | | | | | | | |

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Cohesion | | | Degree of interconnectedness/closeness btw nodes in a network, and how strongly related and connected the nodes are to each other.  Highly cohesive networks exhibits robustness, where failure of an individual node has a limited impact on overall connectivity. Low cohesive networks are vulnerable to disruption  Node level measures of cohesion: Local centrality (betweenness, closeness, degree, eigenvector, pagerank, Katz), Local Clustering Coefficient, Local Density  Network-level measures: Global centrality (betweenness, closeness, degree, …), Global Clustering Coeff, Global Density, Transitivity, Reciprocity, Average shortest path length, Eccentricity | | | | | |
| Transitivity | | | If a b and b c, then a c. Transitivity is a measure of likelihood of nodes in a graph to from closed triplets, where each node is connected to the other two = T = | | | | | |
| Reciprocity (node, network) | | | Only for directed graphs. Reciprocity = ratio of num of edges which are bi-directional, over total edges in graph = R = | | | | | At node level, reciprocity = num of reciprocal pairs it is involved in = R(v) = |
| Density (network) | | | Ratio of num of edges in network over max possible edges. Max edges in undirected graph = N(N-1)/2. Max edges in directed = N(N-1)  Perfectly connected network = clique = density of 1 or 100%  Can be used as indicators of collusion when observed densities of a subgraph are much higher than the avg density of the entire graph | | | | | |
| Clustering coeff | | | Local clustering coeff = measure of degree to which nodes in a graph tend to cluster tgt, or transitivity [0, 1]  Measure of num of links btw nodes within its neighbourhood, divided by num of links that could possibly exist btw them  Clustering coeff of node i = , where ejk is list of edges that connect vertex j and k. And vertex j and k belong to the set of neighbours of i. And ki = num of degrees of i  Global clustering coeff = Mean of local clustering coeff across all nodes [0, 1] | | | | | |
| Avg Longest Distance, Diameter (network) | | | | | | Shortest path can be found via Dijkstra's algo  Network's diameter = longest shortest path btw any 2 nodes in a network  Diameter is useful for indicating reach of a network. E.g. 7 degrees of connection separates the whole word  Avg of all shortest paths indicates the "average dist" btw how far apart 2 nodes will be | | |
| Small World graph + Preferential Attachment | | | A colorful network of balls  Description automatically generated with medium confidenceType of graph w high clustering coefficient, and low distances  Preferential attachment = the more connected a node is, the more likely it is to receive new links  Nodes w higher deg have stronger ability to attrack links added to the network  Is a property of some types of networks where majority of new edges are added to nodes w an already high deg, causing deg of these central nodes to incr disproportionately  Visible in low-tailed deg dist, tend to have small world structure | | | | | |
| Node classification | | | | Suppose network have some nodes whose labels are known, and some unknown. If network is homophilic, how to predict labels of unknown nodes? 1) Relational Neighbor (RN) Classifier. 2) Probabilistic RN (PRN) Classifier. 3) Naive Bayes RN. 4) Decision Tree RN | | | | |
| RNC | | | Estimates class probs based on entities of the same type, whose class labels are known, and does no learning  Node's posterior class prob of belonging to class c: where  neighborhoodn = neighborhood of node n, w(n, nj) = weight of edge btw n and nj, Z = normalization factor to make sure probs sum to 1 | | | | | |
| PRNC | | | Extension of RNC, estimating probs as the weighted mean of the class-membership prob of all entities in that network  Node's posterior class prob of belonging to class c: , where P(c|nj) = result of a local model, or previously applied network estimate. Note summation now ranges over the entire neighborhood of nodes | | | | | |
| Comparison | | | RN, PRM (probabilisitc relational model), RPT (relational prob trees), RBC (relational Bayesian classifiers)  All model perform equally well as % of labelled nodes incr. Challenges was in performing w sparsely-labelled dataset  PRM can get close to its best performance, even w 5% of labelled data. RN competitive w RPT, and out-perform RPT at 5% of data  RN outperform RBC. PRN perform better when very few labels are known, and when there is high skewness in class labels | | | | | |
| Community | | | Community detection = discovering groups in network where groups are not explicitly given = finding groups of closely-connected nodes to identify subgraphs. Fraud tends to occur at the same nodes (identify communicty. of fraudsters). Type of community detection:  1) Node-centric community: Each node in a grp satisfies certain properties. Use cliques/k-cliques OR CPM (Clique percolation mtd)  2) Group-centric: grp satisfies certain properties not present at node-level. Use DBSCAN (Density-Based Spatial Clustering of Apps w Noise)  3) Network-centric: Entire graph partitioned into several disjoint sets by clustering. Use clustering based on vertex similarity OR latent space models OR Spectral clustering OR Modularity maximisation  4) Hierarchy-centric: Hierarchical structure. Use divisive clustering OR agglomerative clustering | | | | | |
| 1a) Cliques | | | Subset of nodes within a graph, s.t. every pair of nodes is directly connected by an edge  1-node cliques = node. 2-node = edges. 3--node = triangles. 4-node and above = polygons  Finding clique of n-size in graph is NP-complete, but also NP-hard (NP = nondeterministic polynomial) | | | | | |
| Maximal clique = clique that cannot be extended by adding an additional adjacent vertex; clique that is not a subset of a larger clique  Maximum clique = largest complete subgraph within a network; largest of all maximal cliques  Finding maximum clique is NP-hard w several approx mtds: a) Brute force: check all possible subsets of nodes to see if they are a clique  b) Bron-Kerbosh Algo: recursively explores graph while maintaining 3 sets of vertices (curr clique, potential nodes to add, excluded nodes)  c) Branch & Bound algo: prune nodes that comfirm won't lead to maximum clique until graph is small enough, then search for max clique | | | | | |
| Adjacent K-cliques: 2 cliques are adjacent when they share (k-1) nodes. Adj cliques are considered 1 community | | | | | |
| 1b) CPM | | | Connected Components = subgraph in which any 2 nodes or cliques are connected to each other, and not to any additional nodes in the supergraph. Node w no edges is a component. Graph w all nodes connected = 1 component  CPM identify overlapping clusters/communities, based on a pre-defined percolation threshold  Community = maximal union of k-cliques that can be reached from each other through a series of adj k-cliques  1) Set param k = desired percolation threshold. 2) Find list of all cliques of size k within graph G. 3) Construct a clique graph Gc, where each node represents an identified k-clique, and adjacent cliques are joined by an edge. 4) Each connected component (joined by an edge) = community. 5) Set C will be the set of communities formed in G | | | | | |
| 2) DBSCAN | | | Density-specific mtds find clusters of nodes w high internal density in terms of attr similarity (NOT the same as density of connection btw nodes). Mtds: K-means clustering, DBSCAN (auto selects clusters, robust to noise),  OPTICS (Ordering Points To Identify the Clustering Structure; extension of DBSCAN, slow for large datasets),  Mean-Shift clustering (non-parametric algo that identifies modes; sensitive to choice of bandwidth params),  GMM (assigned clusters based on prob of belonging to each Gaussian dist; allow for soft assignment, but sensitive to num of components selected)  Affinity Propagation: use message-passing to find exemplars (nodes most representative of their clusters; auto selects best num of clusters, well-suited for graph data) | | | | | |
| DBSCAN: All data fall into 3 categories: 1. Core Points (shape body): Data point is core point if there are ≥ n data points (including itself) within a neighborhood dist of from it  2. Border points (shape outline): Points that are not core points, but still within dist of a core point  3. Noise points (outliers): neither a core point nor a border point  Algo: Identify core points -> Pick a core point, cluster those within -dist of it as same cluster -> Identify all clusters -> Identify border points -> Rest is outliers | | | | | |
| Pros: Highly effective for irregular cluster shapes. Auto determines num of clusters based on density of data. Robust to presence of noise and identifies outliers. Effective in high-dimensional datasets. Efficient for large DBs as it focus on local density rather than global dist.  Flexible parms settings (neighborhood dist , min density per region n). Less sensitive to initial config and random selection of points  Cons: Sensitive to choice of params (neighborhood dist , min density per region n). Sensitive to choice of dist metric. Uneven clustering on regios w diff densities. Computationally expensive on very large datasets to find pairwise distances (use HDBSCAN instead) | | | | | |
| 3a) Clustering on vertex similarity | | | | | Similarity of node's neighbborhood. Structural equivalence = 2 nodes are connected to the same set of actors  Jaccard similarity = . Cosine similarity = | | | |
| 3b) Ratio & Normalized cut | | | Cut = partition of nodes of a. graph into 2 disjoint sets. Problem is with imbalance partition  Community detection = minimum cut problem: Find graph partition s.t. num of edges btw 2 disjoint sets is minimised  For best partition, can use OR , where  = sum of weights of edge(s) that's been cut, |Pi| = num of nodes on each side of the cut, vol(Pi) = sum of deg of nodes on each side before cut, k = num of communities after the cut. Lower the value, more balanced the cut | | | | | |
| 3c) Spectral Clustering | | | Variant of clustering algo that uses connectivity (instead of compactness) btw data points to form clustering  Uses eigenvalues and eigenvectors of the data matrix to forecast data into lower dimensions space to cluster the data points  Based on idea of graph representation of data where data pts are nodes and similarity btw data pts are represented by edges  1) Form a dist matrix btw nodes (dist can be euclidean, cosine similarity, ...)  2) Transform dist matrix into an affinity matrix A. 3) Compute degree matrix D and the Laplacian matrix L = D - A  4) Find eigenvalues and eigenvectors of L. 5) W eigenvectors of k largest eigenvalues, form a matrix  6) Normalize the vectors. 7) Cluster data points in k-dimensional space | | | | | |
| 4a) Divisive Clustering | | | Hierarchical clustering groups data tgt that are close to each other based on measure of similarity or dist  Assume data that are close to each other are more similar or related  Flexibility to choose num of clusters allows us to analyse network at diff granularities. Clusters = communities  Types: 1) Agglomerative clustering (bottom-up): Start w each data point as its own cluster, then aggregate them as dist measure decr  2) Divisive clustering (top-down, splitting): Combine all data points as a single cluster and divide into smaller as dist btw them incr | | | | | |
| Recursively partition nodes into increasingly smaller sets, until network reaches predetermined num of clusters.  Each connected component forms a community. Can use metrics like high edge betweenness centrality, standard square error (SSE), ...  Computationally efficient and good for large datasets and clustes. But not commonly used | | | | | |
| 4b) Agglome-rative Clustering | | Inputs: proximity matrix, dist metric, linkage fn, num of clusters. 1. Compute proximity matrix using some dist metric  2. Use linkage fn to group objs into a hierarchical cluster tree based on computed dist matrix  3. Data points w close proximity are merged tgt to form a cluster. 4. Repeat steps 2 and 3 until single cluster remains | | | | | | |
| Proximity matrix = matrix consisting of dist btw each pair of data points. Dist can be computed using Euclidean, Jaccard index, cosine...   |  |  |  |  | | --- | --- | --- | --- | | Linkage | To measure dist btw 2 clusters: use < X > pairwise dist btw elements in each pair of clusters | Tends to produce ... | Noise | | Complete | X = max | compact, spherical clusters |  | | Single | X = min | elongated clusters | sensitive to noise | | Average | X = average | Compromise btw complete and single linkage | Less sensitive to noise |   d) Centroid linkage: Before merging, dist btw 2 clusters' centroids are considered. Tends to produce well-balenced clusters and is sensitive to overall dist of data  e) Ward's Mtd: Use squared error to compute similarity of 2 clusters for merging. Minimizes incr in var within the clusters when merging. Produce compact and spherical clusters, and used when aim is to minimize var. Compared to complete linkage, this is more computationally expensive due to need to update cluster centroids per iteration | | | | | | |
| 4) | How to know what is ideal num of clusters/communities? By domain knowledge, look at dendrogram, search for elbow improvement, calculate silhouette score, trial and error  Difficulty handling large clusters due to high time and space complexity. For computing proximity matrix, time complexity = O(N2), and since it takes N steps to search, total time complexity = O(N3)  Divisive algo more be more accurate, since agglomerative considers local patterns w/o initially taking into account of global dist of data | | | | | | | |
| Agglomerative Pros: Simple implementation, supported in python libraries  - Natural representation of hierarchy makes it easier to understand data structure and find insights - No need to specify clusters  - Robust to initial config (since all points start as single cluster)  Cons: Computationally intensive - Poor performance on large datasets | | | | | | Divisive pros: Computationally efficient  - Flexibility of num of clusters, although range must be pre-specified  Cons: Complexity of implementation, no standard mtd or library  - Sensitive to initial config (how to split clusters effectively)  - No natural representations of hierarchy  - Diff in defining splitting criteria | |

|  |  |  |
| --- | --- | --- |
| Payment Fraud | Any fradulent or deceptive activity that aims to obtain funds or valuables through illicit means during a financial transaction  Involves manipulating payment systems or exploting vulnerabilities to gain unauthorized access to funds or sensitive info  Payment fraud can occur in various forms and across diff channels: Credit/debit card payments, Wire transfers (bank transfers), Cheque, Cash & counterfeits | |
| Card Payment Methods | Grouped into 1) Card Present (CP). 2) Card Not Present (CNP)  1) CP transactions occur when the physical credit card is present at point of sale, typically where cardholder hands over their card to merchant (eg., PayWave, swiping a card’s magnetic strip, inserting the card, or tapping/waving a card on a contactless POS reader).  2) CNP transactions occur when the credit card is not physically presented during the payment process, often in online or over-the-phone transactions (eg., PayNow, ibank transfer). | |
| Card Payment Ecosystem | The Card Issuing Ecosystem Explained - Carta Worldwide- Merchant uses a payment gateway, a software on a POS machine or online checkout page, to securely communicate w the acquiring processor.  Eg: Paypal, Adyen, Stripe  - Card networks, aka interchanges, identify fraud scenarios and behavioral rules, determine acceptable risk thresholds and enforce policy standars (eg, AML, anti-terrorism financing) on behalf of all its network participants  - Issuing bank and its processor also can reject transactions.  - Issuer may sometimes decide to approve low-risk transaction w incorrect details for the cardholder's convenience. E.g. CVV2 is wrong, but transaction is low amt and to a low-risk merchant that's in line w your past history, issuing bank may still allow it to proceed  - Industry SLA for a settlement decision over the entire transaction's component must be ≤ 200 milliseconds for 99.9% of all requests | |
| Payment Status Code | One transaction can be linked to several payment statuses at diff timestamps, and most recent timestamp indicates its current status.  Statuses: PENDING, SUCCESS, FAILURE. Impt codes: INITIATE/AUTHORISE = Performed the 1st time a card is used.  CAPTURE = Captures and 'reserves' the amt charged so it cannot be used for other transactions.  DENIED = Issuer/network blocks transaction. SETTLEMENT = Withdraw the captured amt.  CANCELLED = merchant or customer initiates cancellation of transaction before settlement is reached.  - While merchants of physical goods would enact both CAPTURE and SETTLEMENT instantaneously upon purchase, merchants providing services may choose to enact them separately, and hold the amt in escrow until service has been completed to customer's satisfaction  https://www.curopayments.com/docs/api/?request\_TransactionCodes | |
| Financial Conseque-nces of Fraud | 1) Loss of Merchandise/Services rendered: Merchants are commonly held responsible for fraudulent transactions with CNP fraud. Chargebacks can result in reversal of funds, leading to direct financial losses for merchant if the goods/service has already been delivered.  2) Chargeback Fees regardless of outcome: Chargebacks, which occur when customers dispute transactions due to fraud, often result in a  chargeback arbitration processing fees for merchants (unfortunately, even if they win the dispute).  3) Increased Processing Fees per Transaction: High sustained chargeback rates may lead to poorer credit ratings for the merchant, which leads to increased processing fees, platform-specific penalties, and even the potential for termination of merchant accounts completely.  4) Higher Authentication & Defence Spending: To mitigate the risk of fraud, merchants may need to invest in advanced security tech and implement additional fraud prevention measures. This can lead to increased operational costs and impact profit margins. | |
| Operational Conse-quences of Fraud | 1) Operational Disruption: Dealing with fraud incidents and chargebacks can be time-consuming and may divert resources  from core business operations. Merchants may need to allocate time and staff to investigate fraud cases, respond to chargeback disputes, and implement additional security measures.  2) Strain on Customer Relationships: Customers who fall victim to fraud may hold the merchant accountable, even if the fraud occurred due to factors beyond the merchant's control. Managing customer relations becomes challenging when customers perceive the merchant as not adequately protecting their sensitive information.  3) Reputation Damage: Experiencing frequent instances of fraud can damage a merchant's reputation. Customers may lose trust in the security of the merchant's payment processes, leading to a decline in customer loyalty and potential negative reviews.  4) Increased Compliance Requirements: Merchants may face heightened regulatory scrutiny and compliance requirements in response to  high fraud rates. Regulatory bodies may impose additional security measures or demand improved fraud prevention protocols. | |
| Payment Protection | To verify cardholder is authentic: 1) Card Verification Value (CVV). 2) Address Verification Service (AVS). 3) 3D Secure (3DS). 4) Device Fingerprinting. 5) Know Your Customer (KYC). 6) ML Solns  1) CVV code = security feature for CNP transactions, and now appears on most (but not all) major credit and debit cards  - 3 or 4-digit code which provides a cryptographic check of the info embossed on the card.  - Visa calls it CVV, MasterCard calls it Card Validation Code, CVC, AMEX calls it Card ID, CID.  2) AVS code tells whether address declard in the order actually matches cardholder. 4 diff ways to do an address check  - PARTIAL\_STREET: street num matches [HIGH RISK]. - PARTIAL\_ZIP: ZIP code matches.  - PARTIAL: Both street num and ZIP code partial matches. - FULL: Only exact street num and zip code matches [LOW RISK]  3) Device Fingerprinting: analyze unique characteristics of device used to initiate a transaction, such as IP addr, browser settings and hardware specs. By comparing info against known patterns of fradulent activity, can help identify suspicious transactions  - Many bank apps will not allow themselves to be activated on a device that has a suspicious fingerprint (e.g. malware, virtual emulator)  4) 3D Secure (3DS): Additional layer of security for online credit and debit card transactions. Adds an authentication step, requiring cardholder to enter a password, or one-time-code or approve a push notification to verify identiy. A form of MFA.  - Regulation requires all transactions made for cardholder accts based in SG must be authenticated w 3DS.  5) KYC = mandantory process of identifying and verifying client's identity when opening an acct and periodically over time.  - KYC standards are designed to protect financial institutions agains fraud, corruption, money laundering and terrorist financing  - In SG: financial instituitions need to verify Full name, Alias, Identification Number, Residential Addr, DOB, Nationality, Independently verified phone num, Telephon confirmation of employment status (w consent)  6) Advanced fraud detection systems use ML algos to analyze and detect suspicious activity in real-time  - Can identiy unusual spending behavior, high-risk transactions. E.g. rule-based detection systems, ML classification algos | |
| ML in fraud detection | - Data science team receives an alert from a dashboard about a spike in fraud losses. What is the cause and pattern of this fraud behavior? How to explain this pattern to impacted merchants, and rebuild their confidence in us? What are some fraud rules to stop this urgent problem in the short term, and what are the precision and recall rates for each? What should we advise our senior management in terms of appropriate risk thresholds for these rules?  - Given a transaction, create a model to determine its risk score, or a measure of how likely it will be classified as fraudulent. Should the team build a scoring model based on the lone transaction, or based on the customer account, or hybrid? How frequently should the model be updated? What if a customer’s account is hijacked by an intruder, how long should their scores be impacted, and should we develop a separate model for hijack prediction? What is the error margin tradeoff for batching or in streaming this service, and will the team be able to deliver within 200ms?  - Suppose the product team builds an additional functionality to ‘escalate’ or ‘challenge’ a risky transaction in the form of an OTP, or push notification, or a request for an additional PIN for verification. However, this should be used sparing as it causes friction, delays, and incurs overhead costs. The team now has to classify transactions as fraudulent, not fraudulent, or ‘to escalate’. How would you build this model in a way that merchants and other stakeholders can customise their desired risk thresholds?  Some features to consider: Whitelists & blacklists, Velocity dashboard, Authentication Rules, Risk tolerance/scoring models by thresholds | |
| Fraud Patterns | A diagram of a bank account  Description automatically generatedPOS cloning, Card Skimming, Theft, BEC, Wire fraud, Phishing, Account takeover, BIN attacks, Carding/MITM  1) Hypothesis design for investigative analysis during a surge of chargeback cases  2) Fraud authentication rule creation  3) Feature engineering for fraudulent transaction classification  4) Communication and expecation management  Chargebacks: process that allows card holder to dispute a transaction and request a refund from the card-issuing bank  - Can initiate for billing errors (duplicate charges, overcharged), dissatisfaction w purchased goods, unauthorised transactions (i.e. fraud)  Chargeback process:  1) Cardholder initiates request for chargeback to issuing bank, submits chargeback reason code and paperwork  2) Issuing bank investigates claims through its internal fraud analytics process: - Bank consider factors like geolocation, IP addr, cardholder's recent payment history. - [optional] Issuing bank invites merchant to respond to dispute (provide chat logs, additional evidence).  - [optional] Further arbitration by Issuing Bank btw cardholder and merchant  3) Decision reached in 3 days - 5 months: - Success = cardholder receives disputed amt. - Failure = cardholder don't receive disputed amt.  - In both cases, metchant pays non-refundable chargeback fees to cover investigation | |
| 1) POS Cloning | Fraudster obtains a functional POS device from an acquirer or agent while posing as a merchant, or from online resellers/auctions/stealing.  Program the POS devices with the credentials of a legitimate merchant. Use the cloned POS devices to issue returns and gift cards  Refunds and gift cards are quickly redeemed, resold or laundered for profit | |
| 2) Card Skimming | Unauthorized capture of electronic transaction data, typically from debit/credit card transactions, through use of illegal modifiaction devices placed over ATMs and POS machines | |
| 4) BEC | Business Email Compromise (BEC) = scammer uses email to trick someone into sending money or divulging confidential company info. Culprit may pose as a trusted figure, then asks for a fake invoice to be paid, or for sensitive data they can use in another scam | |
| 5) Wire Transfer | Fradulent attainment of banking info to gain access to another person's bank acct, through electronic communication mechanisms instead of face-to-face communication.  - Scammers send phishing messages and demand payment via wire transfers. - Scammers may infiltrate legitimate conversations online and manipulate valid payment info. - Conversations w attorneys, real estate agents, lawyers and hospitals tend to be infiltrated due to large amts of money. - After payment has been transferred, extremely diff to recover funds  - Everything about transfer will appear genuine, but recipient acct details will differ. | |
| 6) Phishing | Malicious actors send messages pretending to be a trusted person or entity. Phishing messages manipulate a user, leading victims to perform actions like clicking a link, or divulging sensitive info.  - On Path attack: through malicious links or downloads, fraudsters place themselves btw 2 devices to eavesdrop info, or impersonate either of the 2 agents  - Cross-site scripting (XSS) attack: fraudsters create a realistic looking website w malicious javascript code that collects unauthorised info | |
| 7) Account Takeover | Form of identity theft and fraud, where a malicious 3rd party successfully gains access to a user's acct credentials. By posing as the real user, criminals can change acct details, send out phishing emails, steal financial info or sensitive data, or use any stolen info to further access accts within the organization.  - Hackers may use phishing links to perform an acct takeover OR social engineering techniques to mislead victims to revealing their pw | |
| 8) BIN attacks | A close-up of a bank  Description automatically generatedAct of guessing an accurate combination of the following using brute force computing:  1) Debit or credit card num. 2) Card Verification Value (CVV). 3) Expiry date  Num of combinations: - Credit cards are valid for 5 years = 12 \* 5 = 60 combinations of MM/YY. - 3 digits CVV = 999 combinations  - Acct num = 9 digits = < 1B combinations. - BIN number = public knowledge, fixed. - Last digit is a checksum via Codabar system  Codabar Check: 1) Add the digits in the odd positions and double the total.  2) Add the digits in the even positions  3) Count the num of digits in the odd positions that are > 4.  4) Add the 3 nums obtained in 1), 2), 3)  5) The check digit is the num needed to bring this total up to the next multiple of 10  Codabar is 1 of the most efficient error detection mtds. Picks up 100% of all single digit errors, and most other common errors such as switching 2 adjacent digits. | |
| Signs of Payment Fraud | POS cloning – Unusual voucher credit & refund velocity, maybe to newly created accounts. Merchant has disabled rate limiting or has no regard for chargeback fees.  Card skimming – A small CP transaction, followed by an extremely large CP one  Theft – A small CP transaction, followed by an extremely large CP one. Additional, large CNP transactions may be filed in an international locale that does not require 3DS.  BEC, Phishing, Account Takeover – A large sum is deposited into an acquiring bank account whose details are new and unusual, often in a locale with more lax KYC requirements  Wire fraud – Multiple small transactions looking to sneak by; foreign location + no 3DS authentication  BIN attack – Multiple transaction attempts from the same host. Multiple transactions of incr or decr total value until one gets approved.  Others – Charges for a service, but amt is not CAPTURED, it is immediately debited. Subscription renews at odd intervals, like daily instead of weekly/monthly (which is not financially sound because of the tradeoffs in processing fees, user friction and administrative costs). | |
| Unbalanced Datasets | Imbalanced dataset = num of obs varies btw the classes. Might produce bias, leading to unrealiable results  Resampling can be effective for ML algos that are affected by a skewed dist and algos that learn coefficients, like ANN that use SGD  Can also affect models that seek good splits of data, such as SVM and DT  Might be useful to tune the target class dist. In some cases, seeking a balanced dist for a severly imbalanced dataset can cause affected algos to overfit the minority class, leading to incr generalization error. | |
| Metrics | For balanced datasets, ok to use Error Rate as a metric (1 - Accuracy)  In unbalanced dataset, predicting all unknown inputs as majority class will lead to high accuracy but is not helpful. Better to use ROC curve, or by precision and recall.  Accuracy = (TP + TN) / (TP + TN + FP + FN) | Sensitivity = TP / (TP + FN)  Specificity = TN / (TN + FP)  Precision = TP / (TP + FP)  Negative Predictive Value = TN / (TN + FN) |
| Rebalancing Datasets | |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | | Data-level | Oversampling | Random Over Sampling (ROS) | SMOTE | Borderline SMOTE | ADASYN | | Undersampling | Random Under Sampling (RUS) | Cluster Centroids Undersampling | | Tomek Links | | Hybrid sampling | RUS + Oversampling | RUS + SMOTE | RUS + Borderline SMOTE | SMOTE + Tomek | | Algo-level | | Cost-Sensitive | Ensemble | Tree algo |  | | Hybrid | | Deep-SMOTE | AD-SMOTE | CNN-SMOTE | SMOTE-BOOST | | |
| ROS = randomly duplicating examples from the minority class and adding to training dataset.  RUS = randomly selecting examples from the majority class to be excluded from the training dataset  - Risks: vast quantities of data are discarded. Can make decision boundary btw minority and majority instances harder to learn | |
| SMOTE (Synthetic Minority Over-sampling Technique): generates synthetic samples for the minority class.  Synthetic Sample = Original Instance + (Neighbor Instance - Original Instance) \* Random Value  Problems: SMOTE don't have any jitter, so outlier points will sometimes match up w points inside more dense regions of the class cluster. These connections get interpolated into long "data lines" which may not represent the nature of the minority class well | |
| A diagram of a diagram  Description automatically generatedBorderline SMOTE: Oversampling only on the borderline of the minority class  Focuses on instances near the decision boundary btw 2 or more classes, which are ambiguous and likely misclassified. Can give higher resolution to problematic areas.  All minority points are sorted into: 1) Noise = all m nearest neighbors are from diff classes. 2) Danger = half or more m nearest neighbors are from a diff class (i.e. near the border). 3) Safe = all m nearest neighbors are from the same class  Borderline-SMOTE 1 randomly selects a few types of samples from the KNN sample during new SMOTE for Danger, similar to SMOTE  Borderline-SMOTE 2 pick out any sample from any type of group in the KNN, regardless of sample category  SMOTE 1 select point from the m nearest points NOT belonging to the given point's class  SMOTE 2 select point from the m nearest points of any class | |
| ADASYN (Adaptive Synthetic Sampling) = generating synthetic samples inversely proportional to the density of the examples in the minority class, as an extension of SMOTE. Generate more synthetic examples in regions of the feature space where the density of minority examples is low, and fewer or none where the density is high  1) Calculate the degree of class imbalance, *d = ms / ml,* where *ms =* num of minority instances and *ml =* num of majority instances  2) Calculate number of synthetic samples required for the minority class as whole, *G = (ms - ml) x ß where ß* ∈ *[0, 1]*  *- ß* is the desired balance level after the generation of synthetic data. E.g.: If we desire a 1:1 ratio, then *β*=1  - Note: Round the value of G to a whole number, since we can’t generate fractional samples.  3) For each minority class obs xi, identify its K-nearest neighbors from both the minority and majority classes  - Use euclidean distance within the n-dimensional space. Value of *K* is typically chosen based on a specified parameter. Default = 5  4) For each minority class obs xi, compute density ratio rias the ratio of minority class instances to majority class instances among its KNN: - ri = ∆i / K, where ri∈ *[0, 1], ∆i =* num of neighbours belonging to the majority class, *K* = num of neighbours/parameter chosen in step 3  5) Compute the relative contribution of xiby normalising ri according to a density distribution , where  6) Calculate the number of synthetic examples needed to be generated: *gi = x G*  *-* where *gi* = num of synthetic examples generated for said minority instance, = density ratio for a minority class instance, from step 5  *- G* = num of synthetic samples required for the minority class as whole, step 2  7) For each minority obs xi*,* randomly select a neighbour xjto generate a synthetic sample with, until *gi* num of samples are created:  - Create a sample by selecting a random point along the line segment btw xiand neighbor xj.  8) Repeat until all minority instances have been looped and samples have been created for each. | |
| From left to right: Original. ROS. SMOTE. ADASYN. ROS generates multiple 'copies' of minority instances w/o transformation  SMOTE generates synthetic samples btw a random minority point and its minority neighbor along a straight line  Borderline-SMOTE generates more samples for "danger" points near the "border" btw classes (not shown)  A collage of images of different colors  Description automatically generatedA collage of images of different colors  Description automatically generatedADASYN generates more samples for regions of low minority sample dentiy (near the "border" or otherwise) | |
| A screenshot of a diagram  Description automatically generatedCluster Centroids Undersampling (CCUS): remove instances of low importance from the majority class  - cluster centroid is found by obtaining the avg feature vectors for all features, over the data points belonging to the majority class in feature space. (can use Euclidean dist)  - The instance belonging to the cluster (majority class) which is farthest from the cluster centroid in feature space, is considered to be the most unimpt instace. Least impt instances are removed. | |
| A diagram of a cell  Description automatically generated with medium confidenceTomek Links Undersampling: Remove instances from majoriy class that are close in dist to a minority class, and no other  - Pts a and b define a Tomek Link if (i) pt a;s nearest neighbor is b, (ii) pt b;s nearest neighbor is a, and (iii) a and b belong to diff classes  - All majority class instances in a tomek-linked pair will be removed | |
| Hybrid Sampling: combining multiple sampling strategies. Often, first apply oversampling technique, then apply undersample.  Popular combi: SMOTE + ENN, SMOTE + Tomek Links, RUS + SMOTE  Pros: combi of both leverages the strengths of both techniques, while mitigating their drawbacks | |
| Best mtd is 2004 paper in decr order: 1) SMOTE + ENN, 2) ROS, 3) SMOTE + Tomek  ROS surprisingly decent amongst oversampling mtds. Pruning rarely leads to improvement in AUC for both original & resampled datasets. | |

|  |  |  |
| --- | --- | --- |
| NLP | | Common tasks: Named Entity Recognition (NER), sentiment analysis, translation, text summarization  Can use NLP to detect fraud in BEC, wire fraud, phishing via email, account takeover |
| Preprocessing: Tokenisation & Noise Removal (stopwords, remove or replace URLs w token, remove punctuation, remove industry specific terms, remove or replace social media entities)  1) Lexicon normalisation: Lemmatization (root word) OR Stemming (remove suffix or prefix; word might not exist)  2) Object standardisation: slang, acronyms, shortened word and modern terms. Manual dict lookup to clean data  3) Lowercase. 4) Replace contractions and abbreviations (Mr. -> mister, don't -> do not) to ensure uniformity and prevent duplicate tokens  5) Spell check. 6) Replace or remove emojis and emoticons |
| Feature Engineering  1) Syntactic Parsing: sentences are composed of words in a given order whose meaning is affected by its context  - Dependency Trees: syntactical analysis that deals w asymmetrical binary relations btw 2 lexical items (words). Every relation can be labelled as a relation, governor or dependent, in the form of a triplet  - POS (parts of speech) tagging: noun, verb, adjective, adverb. Used for word sense disambiguation, improving word-based features, removing stopwords and aims with normalisation and lemmatization  2) Entity Extraction: NER to detect names of persons, locations, companies. Typical NER consists of 3 blocks:  - Noun phrase detection using dependency parsing and POS tagging. - Phrase classification using open databases of names and locations  - Entity disambiguation, to clarify potentially misclassified entities  3) Topic Modelling: identify topics using unsupervised learning, based on repeating pattern of co-occuring terms in a corpus  - Latent Dirichlet Allocation (LDA): clustering algo similar to K-means  4) TF-IDF. TF(t,d) = . IDF(t) = log [ N/(1 + df) ]. TF-IDF(t, d) = TF(t, d) \* IDF(t)  5) Count and Readability features: word count, sentence count, freq of industry-specified terms (manually defines). Readability measures include SMOG index, Gunning Fog and Flesch Reading Ease index |
| Techniques like TF-IDF, BoW are part of statistical NLP, that transform text data into vectors or matrices  Document-Term Matrix = such techniques, which describes the freq of words/phrases occurring as a matrix  - Singular Value Decomposition (SVD) used to perform Latent Semantic Analysis (LSA). LSA used for concept searching, auto document categorisation, text summarisation and other related functions. LSA assumes words close in meaning will occur in similar pieces of text, and a document's structure can be condensed via SVD  - Probabilistic Models, LDA applied in Topic Modelling. LDA uses priors from Dirichlet dist for both document-topic and word-topic dist  Limitations w DTM: - don't capture word order, sequence related, or contextual info ("Let's eat, Grandma" vs "Let's eat Grandma")  - High dimensionality and sparse matrix leads to very long processing times and extraction of inaccurate patterns  - Assumes words are independent of each other, which is seldom true |
| NN for NLP | Using one-hot encoding have problem of curse of dimensionality. Instead use embeddings = dense vector w reduced dimensions  Embeddings = type of word representation allowing words w similar meaning to have similar representation.  - Each word is mapped to 1 vector. Requires large amt of corpus data and repeated occurrences of individual exemplars, and has long training time. However, result is a dense vector w fixed, arbitrary num of dimensions  Word2Vec = statistical mtd for efficiently learning a standalone word embedding from a text corpus  - Uses CBoW and skip-gram for computing vector representations of words.  - CBoW = predict target word from context (several words). - Skip-gram = predicts context from target word  - Input layer = one-hot vector. Hidden layer = linear (identity). Output layer = softmax classifier  Limitations: - cannot capture diff senses of words (context independent). Need to take word order into account  - cannot address out-of-vocabulary words. Use characters or subwords | |
| BERT (Bidirectional Encoder Representations from Transformers).  Solves 11+ of the most common NLP tasks: Sentiment analysis, Question Answering (chatbots, voice assistants), Text prediction (auto complete), Text generation, Summarization, Next sentence prediction, Polysemy resolution (can differentiate words that have multiple meanings) | |
| Advanced Fee Fraud (AFF) | | Victims instructed to pay increasingly large advanced fees to facilitate transfer of large sum of money  - Prize winnings, Help/Gratitude money, Employment fee  Cialdini's Influence Theory: - Presentations and assertions of authority (employer, lawyer, lottery manager). - Use of language that signals urgency, to pressure victims into taking decision-making shortcuts  Ferreira et al, solicitation emails: - authority, social proof, liking, similarity, deception, commitment, reciprocation, consistentcy, distraction  - Commitment, reciprocation, consistency most effective when combined w distraction and detection. - Authority plays a secondary role  Conversation level analysis showed a scammer's initial solicitation email and early follow-ups change over duration of conversation  Early emails place heavy emphasis on scam set-up, and how and why victim should engage. Early emails coded w semantic features about authority, desire, secrecy, urgency. Later emails transition to strategies based on liking, similarity and reciprocity |

|  |  |  |
| --- | --- | --- |
| Explainable ML | Transparency and Accountability. Trust and Adoption. Bias and Fairness Detection. Regulatory Compliance. Debugging and Error Analysis. Domain Expert Collaboration & Validation. Educational Purposes  Interpretable ML (IML): - can be understood by human w/o any other aids, tools or methods.  - straightforward, clear structures to understand how inputs are transformed into outputs (rule-based systems, if-else logic, linear regression, logistic regression, decision trees)  Explainable ML (XML): need additional methods to peer into "black box" models like NN, Random Forest, Deep learning techniques | |
| Interpretability vs Explainability: - understand that it is a trade-off. - model complexity is developing so rapidly that we have sacrificed this for performance. - Highly Interpretable Models: linear & smooth, well defined r/s, easy to compute (decision trees, LR, Logistic Regression)  - Highly accurate models: non-linear r/s, non-smooth r/s, long computation time (Random Forest, NN, CNN) | |
| Type of Explana-tions | 1) Self-explanatory: simple, clear, easy to interpret. - Generate explanations at same time as prediction. - Rule-based systems. - Regression models (logistic, linear), Decision Trees  2) Post-Hoc: additional operations performed after model is built. - LIME, SHAPley values  3) Global explanations: Address how model's predictive process works as a whole  4) Local explanations: Provide info for predictions specific to a single instance | |
| Explanable ML Techniques | 1) Example-driven, similarity-driven: explains prediction of an instance by identifying and presenting similar instances. Similar to nearest-neighbors approaches. Limitations: Error prone  2) Feature Importance Plots: Provides score indicating how useful or valuable each feature was in construction of a given model. Scores for each features are calculated and ranked. The more an attr is used to make key decisions, the higher its relative importance | |
| 3) Partial Dependence Plots (PDPs): Visually demonstrate dependence btw an output variable and set of input variables of interest. Due to perception limits, only 1-2 of the most impt features can be visualised in a single graph. Effects can be linear or non-linear | |
| A graph showing the temperature and temperature  Description automatically generated with medium confidence4) Individual Conditional Expectation (ICE) Plots: Demonstrate dependence btw target variable and input feature (similar to PDP), but visualised dependence of prediction for each sample separately. 1 line per sample so overall trends can be visually inferred. Due to perception limits, only 1 of the most impt features can be visualised at a time. Unlike PDP, heterogeneous r/s are not obscured | |
| A diagram of a graph  Description automatically generated5) Local Interpretable Model-agnostic Explanations (LIME): local approximation of a more complex model. Explanations are locally faithful, but not necessarily globally accurate  - Model agnostic. - Can be applied on almost all formats of data (tabular, text, images)  , where Input = instance of x we wish to understand.  f = complex model, g = simple model, which must be a subset of interpretable models = G (like sparse linear models). = local neighborhood of x. L = loss fn to be minimised  = value to regularise and minimise the complexity of the simple surrogate model (e.g. limit depth of decision tree). TLDR: find argument that minimises loss term while staying as simple as possible  , where = proximity is added to weight the loss according to how close the synthetic datapoint z is to our point of interest, x. Any dist fn can be used  = sum of squared differences btw the complex model's label f(z) and prediction of simple model g(z')  A diagram of a model  Description automatically generated1) Select an instance of interest, x  2) From 1, generate random nearby examples using jitters of original point, x1, x2, …, xk  3) From 2, input these generated examples into black box model, then collect the corresponding outputs to create new dataset, y1, y2, …, yk  4) From 3, adjust the outputs by weighing how close they are to instance x to create an adjusted datasets of outputs, y'1, y'2, …, y'k  5) Select a simple model like linear regression (or DT, hidden markov models), then fit the weighted outputs from step 4 into this model | |
| A graph with numbers and circles  Description automatically generated with medium confidence6) SHapley Additive exPlanations (SHAP): explain how much each features contributed to a prediction, and whether it increased or decreased performance (i.e. what is its average marginal contribution amongst all possible combinations of features)  - To calculate importance of feature j, for each iteration, draw feature values in random order for all features except for feature j, then calculate diff of prediction w and w/o feature j  E[f(x)] = average predicted output for all points in dataset  f(x) = average predicted output for a single instance we selected  SHAP values = how each features contributed to this specific prediction, compared to the average prediction.  A graph with different colored lines  Description automatically generatedE.g. for this selected instance f(x), having a credit\_history\_length = 1 leads to a 1.78% higher likelihood of \_\_(output classification) (i.e., getting a loan approved) than the average point w a similar credit\_history\_length = 1 in this dataset  SHAPley values help us identify which features are important for a given instance, and if they have a positive or negative impact, while feature importance only shows us the global mean contribution of the feature  SHAPley decision plot (RHS): - all instances are plotted as a line  - each line value along the X-axis changes as its feature value changes  - Each segment on the Y-axis is a model feature (default show top 10), where importance is calculated by instances plotted, not by whole dataset  - Part where line connects to the X-axis (top) represent model's output for that instance, f(x)  - Part where line connects to the X-axis (bottom) represents the model's mean output value, E[f(x)]  A graph of a person and person  Description automatically generated with medium confidence- All SHAP values are relative to the model's expected value, like a linear model's effects are relative to the intercept  Beeswarm plot: All instances are plotted as a dot, each feature is plotted as a row, and the feature’s effect on the output is represented by their location along the x-axis. We can see that employee\_is\_future\_manager has the highest contribution (top row on y-axis) (assuming graph is sorted by the default: abs.mean value). Also, we see that if the score for is\_future\_manager is a positive value (ie., ‘yes’), it has a high impact on the output (most red dots are to the right side of the graph), indicating that is\_future\_manager has a high contribution to the output (eg., employee salary).  A close-up of a graph  Description automatically generated  Force plot: This plot shows the magnitude (size of areas) and direction (left/right along x-axis) of each feature contribution arrows or bars. Positive contributions are on the LEFT and negative contributions are on the RIGHT. Stronger contributions are toward the middle of the plot. Useful for showing large number of feature effects clearly in comparison to other plots (default limit: 10). Also useful for visualising multi-output predictions. Also useful for comparing different model performance next to each other because of its compressed format. | |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |