I. Intro

A. Difference between AI, ML, and DL

- AI: Membuat algoritma yang dapat melakukan tugas yang biasanya memerlukan kecerdasan manusia tanpa dibuat secara eksplisit
- Machine Learning: Membuat algoritma yang dapat belajar dari data
- Deep Learning: Membuat algoritma yang dapat belajar dari data dengan menggunakan *neural network* yang memiliki banyak lapisan

B. Difference in AI Methods

- Supervised Learning: Algoritma belajar dari data yang sudah diberikan label
- Unsupervised Learning: Algoritma belajar dari data yang tidak diberikan label
- Reinforcement Learning: Algoritma belajar dari interaksi dengan lingkungan dan mendapatkan *reward* atau *penalty* dari tindakan yang dilakukan

C. AI Pipeline

CRISP-DM: *Cross Industry Standard Process for Data Mining*

- Business Understanding: Memahami tujuan dari proyek
- Data Understanding: Memahami data yang digunakan
- Data Preparation: Menyiapkan data yang digunakan
- Modeling: Membuat model dari data yang sudah disiapkan
- Evaluation: Mengevaluasi model yang sudah dibuat
- Deployment: Menerapkan model yang sudah dibuat

II. Data Preprocessing

A. Data Scrubbing

- Identifikasi data yang hilang
- Menghilangkan outlier
- Memperbaiki format
- Menghilangkan data duplikat
- Menangani data yang tidak konsisten

B. Feature Engineering

- Feature Selection: Memilih fitur yang penting saja
- Feature Reduction: Mengurangi fitur yang tidak penting, dengan contoh menggabungkan fitur yang memiliki korelasi tinggi (jumlah produk A, B, C seluruhnya dijumlah menjadi fitur baru)
- Row Compression: Menggabungkan beberapa baris data menjadi satu baris data, contoh ada item instance untuk tiger, lion, dan bear, maka dijadikan satu baris data saja

C. Data Representation

- One-Hot Encoding: Mengubah data kategorikal menjadi data numerik, dengan setiap kategori menjadi kolom baru dengan nilai 0 atau 1
- Binning: Mengubah data numerik menjadi data kategorikal, dengan mengelompokkan data numerik menjadi beberapa kelompok
- Normalization: Mengubah data numerik menjadi data yang memiliki rentang nilai yang sama

III. Model

A. Training

- Split validation: Memisahkan data menjadi data training dan data validasi (70/30 atau 80/20), perlu randomisasi data sebelum memisahkan
- Cross validation: Memisahkan data menjadi beberapa bagian, dan melakukan training dan validasi sebanyak bagian yang ada
  - K-Fold Cross Validation: Memisahkan data menjadi K bucket, dan melakukan training dan validasi sebanyak K kali, dengan setiap bucket menjadi data tes satu kali dengan bucket lainnya menjadi data training
  - Exhausive Cross Validation: Memisahkan data menjadi semua kemungkinan kombinasi training dan validasi
- Jumlah sampel perlu $\geq 10 \times$ feature

B. Model Evaluation/Post Analyss

Menentukan model baik

- Generalazation Capability: Model dapat memprediksi data yang belum pernah dilihat sebelumnya
- Interpretability: Model dapat dijelaskan dengan mudah
- Prediction speed: Model dapat memprediksi data dengan cepat
- Practicality: Model dapat digunakan jika volume data besar

Error:

- Bias: $y - \hat{y}$, error yang disebabkan oleh model yang terlalu sederhana, performs poorly even on training data
- Variance: $E[(y - \hat{y})^2]$, Error yang disebabkan oleh model yang terlalu kompleks (sensitivity to small fluctuations in the training set)
- Overfitting: Model terlalu kompleks sehingga tidak dapat memprediksi data yang belum pernah dilihat sebelumnya
- Underfitting: Model terlalu sederhana sehingga tidak dapat memprediksi data dengan baik

Correctness and classifiers:

- $precision = \frac{TP}{TP+FP}$, *positive predictive value*
- $recall = \frac{TP}{TP+FN}$, *sensitivity*
- $F1 = 2 \times \frac{precision \times recall}{precision+recall}$, *harmonic mean*
- $accuracy = \frac{TP+TN}{TP+TN+FP+FN}$
- $specificity = \frac{TN}{TN+FP}$
- Jaccard Index: $\frac{TP}{TP+FP+FN}$ or $\frac{|y \cap \hat{y}|}{|y|+|\hat{y}|-|y \cap \hat{y}|}$
- Confusion Matrix: $\begin{bmatrix} TP & FP \\ FN & TN \end{bmatrix}$
- log loss: $-\frac{1}{m}\sum_{i=1}^{m}(y_i \log(\hat{y}_i) + (1-y_i)\log(1-\hat{y}_i))$ Prediction of probabilities

IV. Regression

A. Linear Regression

- Simple Linear Regression: $\hat{y} = \theta_0 + \theta_1 x$ 1 feature
- Multiple Linear Regression: $\hat{y} = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \ldots + \theta_n x_n$ multiple features

B. Minimizing Error

Error Types:

- Mean Squared Error: $\frac{1}{m}\sum_{i=1}^{m}(y_i - \hat{y}_i)^2$
- Mean Absolute Error: $\frac{1}{m}\sum_{i=1}^{m}|y_i - \hat{y}_i|$
- Root Mean Squared Error: $\sqrt{\frac{1}{m}\sum_{i=1}^{m}(y_i - \hat{y}_i)^2}$

Cost Function:

- Mean Squared Error: $J(\theta) = \frac{1}{2m} \sum_{i=1}^{m} (\hat{y}_i - y_i)^2$
- m = number of samples
- Gradient Descent: $\theta_j = \theta_j - \alpha \frac{\partial J(\theta)}{\partial \theta_j}$
- example: $\theta_0 = \theta_0 - \alpha \frac{1}{m} \sum_{i=1}^{m} (\hat{y}_i - y_i)$
- $\theta_1 = \theta_1 - \alpha \frac{1}{m} \sum_{i=1}^{m} (\hat{y}_i - y_i) x_i$
- Least Squares: $\theta = (X^T X)^{-1} X^T y$
- stocahtic gradient descent: Update θ for each sample in the training set, loop through the training set multiple times
- mini-batch gradient descent: Update θ for a subset of the training set, loop until convergence
- Statistic equation $m = \frac{n \sum xy - \sum x \sum y}{n \sum x^2 - (\sum x)^2}$
- Statistic equation $b = \frac{\sum y - m \sum x}{n}$

V. Classification

A. K-Nearest Neighbors

- Desc: Voting berdasarkan k neighbor terdekat
- Hyperparameter: Jumlah neighbor, distance metric (euclidean, manhattan, minkowski)
  - Euclidean: $\sqrt{\sum_{i=1}^{n} (x_i - y_i)^2}$
  - Manhattan: $\sum_{i=1}^{n} |x_i - y_i|$
  - Minkowski: $(\sum_{i=1}^{n} |x_i - y_i|^p)^{1/p}$
- Pros: Simple, no training time
- Cons: Slow prediction time, need to store all data

B. Decision Tree

- Desc: Pohon keputusan yang dibuat berdasarkan fitur yang ada
- Hyperparameter: Max depth, min samples split, min samples leaf
- Pros: Easy to understand, can handle non-linear data
- Cons: Overfitting, sensitive to small changes in data
- Cara kerja, entropy = $-\sum_{i=1}^{n} p(i) \log_2 p(i)$, minimize entropy
- information gain = entropy(parent) $-\sum_{i=1}^{n} \frac{N_i}{N} \times entropy(child_i)$ Maximize information gain. N = total number of samples, N_i = number of samples in child node i
- Gini impurity: $1 - \sum_{i=1}^{n} p(i)^2$, gini lebih simpel, hasil entropy lebih bagus

VI. Logistic Regression

- Desc: Regresi yang digunakan untuk klasifikasi, memberi probabilitas kelas
- Hyperparameter: Learning rate, max iteration
- Pros: Simple, fast prediction time, memberi probabilitas
- Optimize θ for binary classification: $J(\theta) = -\frac{1}{m} \sum_{i=1}^{m} (y_i \log(\hat{y}_i) + (1 - y_i) \log(1 - \hat{y}_i))$
- $\hat{y} = \frac{1}{1 + e^{-\theta^T x}}$ or $\hat{y} = \sigma(\theta^T x)$

VII. Support Vector Machine

- Desc: Mencari hyperplane yang memisahkan data dengan margin terbesar
- Hyperparameter: Kernel, C, gamma
- Pros: Can handle non-linear data, can handle high dimensional data, memory efficient
- Cons: overfitting, no probability, small dataset
- Kernel: Linear, Polynomial, RBF, Sigmoid, akan mengubah data ke dimensi yang lebih tinggi agar dapat handle non linear data. Contoh polynomial kuadrat $\phi(x) = [x, x^2]$

A. Naive Bayes

- Desc: Klasifikasi berdasarkan probabilitas
- Hyperparameter: Smoothing
- Pros: Simple, fast prediction time, can handle high dimensional data
- Cons: Assumption of independent features
- Bayes Theorem: $P(A|B) = \frac{P(B|A) \times P(A)}{P(B)}$
- Naive Bayes: $P(y|x_1, x_2) = P(y|x_1) \times P(y|x_2)$
- Smoothing: Add small value to the probability to avoid zero probability

B. Random Forest

- Desc: Ensemble dari decision tree
- Pros: Can handle non-linear data, can handle high dimensional data, can handle missing data
- Cons: Slow prediction time, hard to interpret
- Bagging: Bootstrap Aggregating, membuat beberapa model dari data yang diambil secara acak
- Random Forest: Membuat decision tree dari subset data dan subset fitur

VIII. Clustering

A. K-Means

- Desc: Mencari centroid dari data
- Hyperparameter: Number of cluster, max iteration
- Pros: Simple, fast prediction time
- Cons: Need to specify number of cluster, sensitive to initial centroid, sensitive to outliers, hasil mungkin beda
- Algorithm:
  - Randomly initialize centroid
  - Distance calculation
  - Assign data to centroid
  - Update centroid position to the mean of data
  - Repeat until convergence
- metric: mean distance between data and centroid
- Elbow method: Mencari jumlah cluster yang optimal

B. Hierarchical Clustering

- Desc: Membuat dendrogram dari data
- Pros: Always give same result, no need to specify number of cluster
- Cons: Slow prediction time, Difficult to identify cluster
- divisive: Start from one cluster and split into smaller cluster
- agglomerative: Start from each data as cluster and merge into bigger cluster
- Algorithm:
  - Calculate distance between data
  - Merge data with smallest distance
  - Repeat until number of cluster is reached
- Dendrogram: Visual representation of hierarchical clustering, can cut the dendrogram to get number of cluster
- Distance calculation: Single Linkage (minimum distance), Complete Linkage (maximum distance), Average Linkage (average distance of all data), Centroid Linkage (distance between centroid)

C. DBSCAN

- Desc: Mencari cluster dari data yang memiliki density yang tinggi
- Hyperparameter: Epsilon, Min samples
- Algorithm:
  - Find core point (data with number of neighbor $\geq$ min samples)
  - Find border point (data with number of neighbor $<$ min samples but in the epsilon distance of core point)
  - Find noise point (data with number of neighbor $<$ min samples and not in the epsilon distance of core point)