PYTHON/DS NOTES

—

Basics

- **Class -** a class has several attributes which you can extract these functions of class to be used in another part of the program or a different program. It is a blueprint or a prototype that tells us what attributes the class contains.
- Python treats everything as an object.
- An object contains specific data and provides functionality specified in class.
- Integer and float objects are **not callable**. Calling an object - create an instance of a class.
- We need self-argument to access all the attributes in the class. It represents the instance of the class.
- **Init method -** a constructor, side effect when a new object is created. Can accept arguments in the init method and store this as an attribute of the object. Def __init__(self,name): self.name = name. Self is the object. Self.name is the attribute of the object.
- **Encapsulation -** combine all attributes of the common class.
- **Abstraction -** shows only necessary attributes, and the rest of the information is hidden.
- **Inheritance -** a parent class has sub or child classes where these subclasses take attributes of the parent class.
- **Polymorphism -** an object can exist in different forms. An object can perform the same function on different data types.
- **Dunders or magic functions -** starts with a double underscore.
    - __init__(self): as soon as an object is created, this method is called with the parameters we created.
    - __del__(self): when the object is deleted out of the memory deliberately.
    - __add__(self, other): whenever we add an object to another object.
    - __string__(self): convert an object to a string. And many more.
- **Overloading** - python does not support it.
- **Overriding** - python forgets the previous value allocated to that object or method.
- **Super** keyword - overriding in the base class. Check more on this.
- **Method resolution order** - C3 linearization: this order defines the order in which multiple inheritances work.

- **Module** - in the case of codes, when we want to share it with people or organizations or other python applications, make a python module. Define a function or class inside the module and import it into another module.
- **JSON** - a common file that is returned from the server.
- Json library -
    - Json.**loads** - load JSON file to string
    - Json.**dumps** - dumping dictionary to string
    - Json.**dump** - dump any other dictionary to JSON file
- **Error** - syntax, runtime (syntax is correct but mathematically may not be like 10/0).
    - **Try and except** - handling errors. Try - try this statement but if this does not work then print or do as per the except block.
    - Both try and except work together.
    - **Raise exception** - custom exception. All errors must raise from the base exception. Can create a class and functions to define custom errors.
- **Mergesort** and **quicksort** - divide and conquer algorithms in python to sort arrays. Both have different logic.
- ValueError - when the type of the argument passed to the function is incorrect.

Probability

- **Multiplicative** rule in probability happens to **dependent events**. Naive Bayes is a good example, along with conditional probability.

Statistics

- Kolmogorov-Smirnoff test is used to find the probability distribution of the data.
- Skewness is the measure of symmetry.
    - Positively skewed: right tail is longer, values more around the left side, median is smaller than the mean.
    - negatively skewed: left tail is longer, values more around the right side, median is greater than the mean. Eg: pareto principle
- Kurtosis: measures the bulgeness of the data. How heavy or light tailed the data is as compared to normal distribution.
    - Mesokurtic: normal distribution no bulgeness
    - Leptokurtic: heavy bulgeness
    - Platykurtic: light bulgeness

- Law of large numbers: It states that if an experiment is repeated independently multiple times, the mean of all results will approximate the expected value.
- The Central Limit Theorem states that the distribution of sample means starts to resemble a normal distribution as the size of the sample increases.

P-Value

- Power of the test is the probability that rejects null hypothesis when alternative is true.
- Hypothesis testing: null hypothesis, alternate hypothesis, experiment, and then accept or reject.
- Significance value or alpha defines the range until we accept the null hypothesis.
-

Class Imbalanced

- SMOTEing

Feature Selection

- Three types - filter, wrapper, and embed method.
- **Filter** method - checks the relevance of the factor with the output variable. For Example, using ANOVA, Chi sq test, correlation coefficient, and information gain.
- **Wrapper** - forward and backward selection method and recursive feature selection. Better for small datasets.
    - In backward selection, we perform a chi-square test to find which feature has the lowest impact on the target variable.
- Embedded method - takes permutation and combination of variables. Example - decision tree.
- Univariate selection - statistical tests used to select features that have the strongest relationship with the target variable. Scikit-learn library: SelectKBest. From sklearn.feature_selection import SelectKBest.
- Feature importance using Extra Tree Classifiers.
- Correlation matrix with heatmap.

Encoding

- One hot encoding for a large number of variables. Take the top 10 variables and consider the other categories as noise.
- Nominal and ordinal encoding: not limited and limited to rank of the categories, respectively.
- Nominal includes one hot encoding and one hot encoding with many categorical and mean encodings.
- Ordinal includes label, target guided.
- Dummy variables trap: take n-1 variables instead of n. This is done because one variable might be used to estimate all the others. High correlation. So we exclude one variable.
- Disadvantage: many categories will increase the number of dimensionalities.
- Label encoding:

Dimensionality Reduction

- PCA or dimensionality reduction is an unsupervised algorithm that helps reduce the number of dimensions or features.
- The first step is usually a standard scaler.
- Why we need to do PCA:
    - Less computing training time taken by the model
    - Avoids the problem of overfitting
    - Useful for data visualization
    - Takes care of multicollinearity
    - Useful in factor analysis
    - Removes noise from the data

Feature Scaling

- Gradient descent-based and distance-based algorithms: features must be on a similar scale for the GD to minimize cost function smoothly.
- Tree-based algorithms aren't sensitive to scaling the features.
- Normalization is a scaling technique in which values are shifted and rescaled between 0 and 1. This is also known as min-max scaling.

- Standardization involves centering the values around the mean with a unit standard deviation. Mean = 0.
- Not a thumb rule but:
    - Normalization - when you don't know the data distribution, like in KNN, Neural networks.
    - Standardization - when data follows a normal distribution. Does not have a bounding range.
- Outliers are not affected by standardization.
- We scale on the training data and then use it to transform the test data to avoid data leakage during the testing process.

Feature Engineering

- For classification:
    - Make the missing rows test data.
    - Make clusters to see in which group the missing row falls.
- MCAR: when there is no relationship between missing data and any other value.
- Temporal variable: DateTime variables.
-

Bias & Variance

- **Bias variance tradeoff:**
- **For Regression problem -**
    - **Underfitting** - just for my training data, the error is very high. This is for linear regression, where the degree of the polynomial is 1. As we **increase** this **degree,** the model will fit the training data perfectly, called **overfitting**.
    - But there is a degree between overfitting and underfitting, giving us low bias and variance.
    - In **Underfitting** - high bias and variance, **Overfitting** - low bias and high variance.
    - **Bias:** a phenomenon that skews the result of an algorithm in favour or against an idea (training dataset). It is the difference between the average prediction of our model and the correct value that we are trying to predict.
    - **Variance** refers to the changes in the model when using different proportions of the training or test data.
    - **Bias** - error of training data. **Variance** - error for test data.
    - **Variance - the** difference in fit between data sets.

    - **Underfitting -** Model not able to learn in the train set. Fix - get more data and features and try other algorithms.
    - **Overfitting -** A model that learns well from training data but not unseen data. Fix - Regularisation, cross-validation.
- **For Classification problem** -
    - Make a confusion matrix in this.

Model 1	Model 2	Model 3
Train error - 1% Test error - 20%	Train error - 25% Test error - 26%	Train error < 19% Test error < 10%
Low bias, high var - overfitting	High bias, high var - underfitting	Low bias, low var - Good model

    - For the decision tree, it is an overfit model - low bias and high variance. We then do decision tuning, forming a model to some depth to minimize overfitting.
    - Random forest - since the combination of decision trees, low bias, and low variance. High variance in one decision tree if combined gives low variance.

Performance Metrics

- When the data is balanced, we use accuracy. If there is an imbalance, we use recall and precision.
- **Confusion matrix** - Across - TP, FP; Down: FN, TN
- FP - **type 1** error. Rejecting the null hypothesis when it is true.
    - This error is called **false positive** ratio or FPR, FPR = FP/(FP + TN).
- FN **- type 2** error. Accepting null hypothesis when it is false.
    - This error is called **the false-negative** ratio or FNR.
- Reduce type 1 and 2 errors.
- For **balanced** problems, directly compute **accuracy** = TP + TN/(TP+FP+FN+TN).
- **Accuracy** is the number of correct predictions made by the model over all kinds of predictions.
- **The recall** is **sensitivity** and **TPR**. Out of the total actual positive values, how many did we predict correctly? R= TP/TP+FN.
- **Specificity** is the exact opposite of recall - TN.
- **Precision** is a function of FP.
- **Precision** is the **positive predictive value**. Precision = TP/(TP+FP). Out of total actual positive values, how many were actual positives?
- When FP is important, use precision; if FN is important, use recall.

- Depending on the model, check if FN or FP is important and then take the corresponding metric for model performance.
- When you want to include precision and recall, use the **F beta** score.
- **F beta** = (1+beta^2) (precision X recall) / (beta^2 X precision + recall).
- If **both** FP and FN are needed, use **beta = 1.** Then the F beta score will be equal to the **harmonic mean.**
- When **FP has more impact** than FN, use a beta value between 0 and 1, usually 0.5. We reduce the beta value in this case.
- **Increase** beta if **FN is more important.** Between 1 and 10.
- **Confidence interval =** point estimate +- margin of error.
- Example: if the stock market is going to crash, focus on both (f-score). For cancer classification, **recall** should be prioritised, and for **spam**, focus on precision.

Tests

- Chi-square test - a non-parametric test performed on categorical variables, nominal or ordinal data.
-

ROC & AUC

- Instead of being overwhelmed by many confusion matrices, **Receiver Operator Characteristic**, ROC graphs **summarise** the information.
- Y-axis is **Sensitivity** = True Positive Rate = True positive/(TP+FN).
- **Specificity is FPR = 1 - TPR.**
- X-axis is False Positive Rate = 1 - **Specificity** = FP/(FP + TN).
- The area under the Curve AUC makes it easier to **compare** two ROCs. A **larger** AUC is better.
- The model should always have an AUC **greater** than 0.5.
- Select a threshold value such that TPR is high and FPR is low.

Linear Regression

- Intercept in straight line equation: intercept tells us at what point we meet the y axis.
- Slope or coefficient: with a unit movement on the x-axis, what is a unit movement on the y-axis?

- Goal is to find the best fit line so the data points are closest to the best fit line. The distance should be minimal. Line equation: h theta (x) = theta not(0) + theta 1 x
- We create a cost function to find the best fit line and the distance between the points.
- Cost function = $1/2n \sum_{i=1}^{n}$ (h theta (x) - y)^2. This equation is known as the mean squared equation. We divide by n to get the average since we iterate through all (n) data points and divide by 2 for easier derivation.
- Adjusted r squared: if we increase independent features, then there is a chance that r squared might increase even if the independent feature is uncorrelated with the dependent variable. Then we use adjusted r squared.
- **MSE** is the mean of squared differences between expected and predicted values. The unit is squared, so the final interpretation square root of MSE is taken (RMSE).
- **MAE**: unlike MSE, the unit matches the target variable's units. It is the mean of absolute errors.
- **R-squared:** how much variation of dependent variable is explained by the independent variable.
- Assumptions:
    - The variables are independent of each other.
    - Data follows a normal distribution.
    - Standardization (scaling data using z-score)
    - Linearity
    - Multi collinearity, solved using Variation inflation factor
    - Homoscedasticity: all variables have same variance
    - No autocorrelation: found in time series data, relationship between each value of errors. Use durbin watson test.

Loss Function

- There are types of loss function, classification, and regression. Below is only **regression**. These are known as linear activation functions.
- **MSE** and **MAE** (L2 & L1 loss) -
    - MAE is useful when **training data is corrupted with outliers**, i.e, huge negative/positive values in our training environment.
    - **Median** is more robust to outliers than mean; hence **MAE is more robust** to outliers than MSE because we don’t square the outlier points in MAE.
    - The disadvantage of MAE - gradient is the **same** throughout; it will be large even for small loss values. We use a **dynamic learning rate** for this. MSE is more **precise** even with a **fixed learning rate**.

    - When to use MSE or MAE:
        - If outliers represent anomalies that are **important** for business - MSE
        - If outliers represent **corrupted** data - MAE
- **Huber Loss** -
    - It is an absolute error, which becomes **quadratic** when an error is small.
    - How small this error makes a quadratic depends on the **hyperparameter**, delta, which can be tuned.
    - Huber approaches MSE when the delta is **approximately 0** and MAE if **infinity**.
    - Residuals larger than delta are minimized with L1 and smaller than L2.
    - Why use - since MAE has a **large** gradient, Huber loss curves around the minima which **decreases** the gradient. More robust to outliers than MSE. **Combines good properties of MSE and MAE.**
    - Problem - we need to **train** hyperparameters delta, an **iterative** process.
- **Log - Cosh Loss** -
    - Smoother than L2.
    - It is the **logarithm** of the **hyperbolic cosine of prediction error**.
    - Works mostly like MSE but is not strongly affected by occasional incorrect predictions.
    - Has all the advantages of Huber loss and is **twice differentiable** everywhere, unlike Huber loss.
    - ML models require the 2nd derivative like XGBoost.
- **Quantile Loss** -
    - When we need to predict an **interval** instead of point-only predictions.
    - It is just an extension of MAE; if it is **50th** percentile, the quantile is MAE.
    - Whether we want to give positive or negative errors depends on the quantile value.
    - For a quantile loss function of gamma = 0.25 gives more penalty to overestimation and tries to keep prediction values a little below the median.
    - Mostly used in neural nets and tree-based models.
- Classification loss functions:
    - **Binary cross-entropy:** use the log loss function as in logistic regression. Find y hat using the sigmoid activation function.
    - Multi-class problem: use relu in the hidden layer and softmax in the output layer.

Logistic Regression

- It is a **binary classification** algorithm.

- The output for this is always a **probability**.
- **Called regression because** it measures the relationship between categorical dependent and one or more independent variables using a sigmoid function.
    - The equation passed to this sigmoid function, consisting of log odds, is linear. Similar to how linear regression is started.
- **Why not use** linear regression for binary classification?
    - Because we need the y predicted to be a probabilistic value, not continuous.
    - Because if we add outliers, the best-fit line will deviate, and the meaning of 0 and 1 will be wrong and give a high error.
- **The math behind logistic regression**:
    - The linear equation for this is an unbounded linear equation. Because the output we need is a probability, it should be between 0 and 1.
    - Hence we use a log of linear equations to bind it using logit transformation. Logit transformation is log(p/1-p).
- The algorithm finds the line with max likelihood by rotating the line to increase the log likelihood.
- **Why need a different** cost function
    - Because the squared error loss is non-convex and has multiple local minima.
- In logistic regression, we **transform** the y-axis from the probability to the y function's **log of the y function.** When we draw a best-fitting line on this, the transformation pushes the data to positive and negative infinity. And this means the residuals are also equal to **positive and negative infinity**. You can't use least squares to find the best fitting line, the coefficients of parameters. So use **MLE**.
- The maximum likelihood equation is not in the exact format. Use newton rapson type of iterative methods to compute coefficients (IRLS method).
- The y variable turns to **the log of odds** of Y.
- The probability is not calculated as the area under the curve but as the **y-axis value,** which is **why it is the same as likelihood.**
- R squared = LL(overall probability) - LL(fit) *whole* / LL(overall probability).
- P-value -
- **Cost function** =

$$-1/n \sum_{i=1}^{n} [y\, log(y_{pred}) + (1 - y)\, log(1 - y_{pred})],\ y_{pred} = \sigma(w^T x + b),$$

  w is the distance of a point from a line, same as a coefficient. Update w_i such that the cost function becomes maximum. Not the same as linear regression because our prediction function is nonlinear due to sigmoid transformation. This cost function represents one observation if for n, take avg of the function.
- The **cost** function is convex and a form of cross-entropy loss classification.
- Use cross-entropy or log loss to calculate logistic regression coefficients.

- **Misclassification rate or Classification error -** how often is the classifier incorrect?
- **Sigmoid function** - a straight line can give wrong predictions. $\sigma(y) = \sigma(mx + c)$, simply taking the sigmoid of the linear equation.
- The mathematical equation of the sigmoid function.
- With n number of features - check the y prediction equation. This gives the probability of the observation being either 1 or 0. If y prediction >= 0.5 then 1 else 0.

Regularisation

- Reduce overfitting and also when there is multicollinearity.
- Overfitting - when data fits in training data but not unseen data.
- If test RMSE > train RMSE, then that model is overfitting.
- Loss function/ Cost function/ lambda? Are all residual sums of squares? This measures the performance of models based on data. The **beta coefficients** are chosen to minimise this loss function.
- Apart from lasso and ridge, dropout regularisation, data augmentation, and early stopping are also used.
- **It reduces the variance of the model without an increase in its bias.**
- **Lambda** controls the impact of bias and variance.
- Lambda is called the regularisation parameter.
- Why lambda should be **carefully selected** - increasing lambda up to a certain limit will reduce overfitting without **losing important data properties**. But after this limit, the data will lose these properties and make the model **underfit**.

Ridge Regression - L2

- If the sum of residuals in training data is minimal**, it is high for testing data**. This means there is a high variance.
- Then we'd say that the new line, for testing, that is, is overfitting the training data. This is by finding the line with the least-squares method.
- In ridge regression, we find a new line that doesn't fit the training data. We introduce a small amount of bias into how the new line fits the data. In return for that small amount of bias, we get a significant drop in variance. Ridge can provide better long-term predictions.
- Ridge tries to **minimize** - Regularization term - the **sum of squares of residuals + lambda * slope^2**, lambda is from - to any positive number.
- We use ridge over linear since ridge has a lesser variance.

- We try multiple values for lambda and use cross-validation to find the one with the least variance.
- As lambda **increases**, the slope decreases and eventually **tends** to 0.
- The coefficient estimates produced by this method are called the L2 norm.
- Disadvantage - model interpretability. It will shrink the coefficients very close to zero but never exactly 0. The final model will include all predictors.
-

Lasso Regression - L1

- Used in the same context as a ridge.
- When slope = 0, lasso = linear.
- **Difference**: ridge only decreases the slope asymptotically close to 0 whereas lasso fully decreases slope to 0.
- Regularization term for Lasso = **sum of squares of residuals + lambda * |slope| - magnitude of the slope.**
- Both don't include the y-intercept in their equation.
- Since lasso can exclude the useless variables from the equation, it is a little better than ridge at reducing variance in models containing many useless variables. **Advantage**: feature selection.
- Looking at its equation below, the lasso has the smallest loss function for all points.
- L1 penalty forces some coefficients equal to 0 when the lambda is sufficiently large. This method also performs **variable selection** and yields **sparse models**.
- Helps in feature selection since it can take coefficients to 0.
- Ridge can be thought of as solving an equation where the summation of squares of coefficients is less than or equal to s.
- Lasso can be thought of as an equation where the summation of the modulus of coefficients is less than or equal to s.
- Example - if there are two coefficients, for ridge - beta1^2 + beta2^2 <= s and lasso, |beta1| + |beta2| <= s.
- Ridge constraint forms a circle and lasso and has corners like a square. That will then let the ellipse cut the constraint at an axis. If this happens, then one of the coefficients will equal zero.

Decision Trees

- A decision tree can contain a mix of numeric and categorical decisions.
- Output is yes or no.

- The very top node is called the **root node**. It is split into **internal nodes**, with arrows pointing towards and away from them. When these internal nodes are not split further, these are **leaf nodes**.
- Which attribute to use to **split** the node? We use **entropy** or **Gini impurity**. Splitting the node quickly is our aim to reach the leaf node.
- Impure node is split until it becomes pure.
- Step 1 - **Induction -** set all of the hierarchical decisions.
- Step 2 - **Pruning -** removing branches that use less important features. Improves the performance of the tree.
- **Pruning** reduces the overfitting and complexity of the tree.
- **Pre-pruning:** stops growing the tree before the training set is classified.
- **Post-pruning:** allows the tree to classify the training set and then reduce the tree. This is a better method than pre-pruning.
- Overfitting for Decision Tree.
- **Cost function** for -
    - Regression tree - Squared Error, $E = \Sigma(Y - \widehat{Y})^2$
    - Classification, Gini Index Function, $E = \Sigma (P_k * (1 - P_k))$, pk is the proportion of training instances of class k in a particular prediction node.
- Classification Trees -
    - Combines numeric data with yes/no data.
    - Numeric thresholds can be different for the same data.
    - Final classifications can be repeated.
    - For numeric - sort the column from lowest to highest. Calculate avg value for adjacent observations. Calculate Gini for each average value.
- **Classification and regression tree** (CART) -
    - Perform **variable screening or feature selection**.
    - Non-linear relationships between parameters **do not affect** tree performance.
    - Can become unstable because small variations in the data might result in a completely different tree. This is called **variance**.
    - The above is lowered by **bagging** and **boosting**.
    - **Biases** can occur if some classes are dominant. Balancing the data is required, then.
    - Regression trees have a numeric value, and classification has true or false in their leaves or some other category.
- **Stopping method** - use a minimum count on the number of training examples assigned to each leaf node.
- **Numerical data -**
    - First, sort the values: in continuous numerical data

    - For DTR: calculate SD for all. This calculates the homogeneity of the sample.
    -
- Which feature to select first for the node? **Information gain.**
- Min sample split, max depth

Entropy

- Information theory - it is based on an intuition that -
    - More likely events give us less information.
    - Less likely events give us more information.
- Measures the **purity** of the split.
- **Entropy, H(S)** = summation -pi log(pi), base is generally 2 = -P+ log P+ - P- log P-, P+, P- is the percent of +ve or -ve class respectively.
- If the split is like three yes and three no, then entropy is **1**. This split is the worst, completely **impure**.
- If the sample is completely homogenous, else if the sample is equally divided, then the entropy is zero else one.
- **Less** entropy is **chosen**.
- Entropy value ranges between **0 to 1**.
- **Gini impurity** = 1 - summation of P square, both positive and negative.
- In entropy max value is 1, and the Gini impurity is 0.5.
- **When to use gini and entropy?** Gini is computationally efficient. Hence we use this more than entropy. Since it has log and gini has simple maths.
- **Information Gain** - it takes into consideration all entropies. To understand which split is better.
- Steps for finding the root/decision node:
    - Calculate the entropy of the target
    - Split the data into different attributes. Calculate the entropy for each attribute.
    - Find the IG of all the above attributes. IG = Entropy before the split - Entropy of attribute.
    - Attribute with the largest IG is the root node.
- **Formula** -
    - IG = H(parent) - summation wi H(child)
    - Gain(S,A) = H(S) - $\Sigma(v\ to\ values)\ |S_v| / |S| * H(S_v)$, where S is the subset Sv is subset after splitting.
- **Highest** information gain is **used**.
- If the split has a high percentage of each class for each input, i.e., the split is not a mix of both classes, then there is a gain in information.

Gradient Descent

- A mechanism that aims to explore a function's minimum value by iteratively moving in the direction of the steepest decrease in the function value.
- An optimization algorithm used to find values of parameters or coefficients of a function that minimises a cost function.
- X-axis is the parameter and y-axis is the loss function value.
- **Learning rate:** this adjusts how much we move at each step. Read more.
- It is a convex function whose output is the partial derivative of a set of parameters of its inputs. We repeat it until we reach convergence.
- What if there is **local minima** in your regression problem when applying gradient descent? Usually, with the gradient descent and cost function, but in deep learning, we face local minima, then optimisers such as adam optimisers and more help solve this issue.
- A cost function is a function that measures the performance of your ML model.
- The first derivation of the cost function, the slope, is calculated to know how to move the coefficient values to get the next iteration cost value.
- Cost of coefficient should be close to 0.
- Common examples - linear and logistic regression.
- Stochastic GD - for a large dataset, this is better.
    - Cost is calculated for each training pattern to give an optimised coefficient value rather than a sum for all at the end (Batch GD).
    - Calculating the derivative from each training data instance and calculating the update immediately.
    - Whereas in Batch GD, the derivative for all training data is calculated and updated. One coefficient update, 2nd coeff update…
- A decreasing cost of iterations is a good sign, if not reduce the learning rate.
- Path taken is noisier in SGD as compared to BGD.
- Hyperparameters in SGD - loss function, learning rate.

Gradient Boosting

- Fundamentals:
    - **Weak learner** is a model that is slightly better than random guessing. Combining multiple weak learners helps in making an accurate model.

    - In GB, weak learner models are added to the ensembles iteratively, **additive models**. This is like a taylor approximation where the final value is predicted using a rough estimate corrected by a series of correction terms.
    - **Loss function:** measures the difference between predicted value and the actual value, evaluating model performance. More loss, worse is the model.
- Hyperparameters:
    - Number of additive models
    - Learning rate
    - Maximum depth of each model
    - Min number of observations in terminal nodes of the weak learners
- Pros:
    - Strong prediction performance: combing lots of smaller models and use the intelligence of these models to make the prediction rather than trying to fit all the data patterns in one model.
    - Flexibility: this can be applied to either regression or classification using different weak learners, loss functions and data types.
    - Interpretability
- Cons:
    - Computational complexity
    - Overfitting
    - Hyperparameter tuning

Cross-Validation

- K-fold -
    - Data is divided into k subsets.
    - Each time, one of the k subsets is used as the test or validation set and the other k-1 as the training set.
- Stratified K fold - according to the classes of the categorical target variable, the stratified k-fold divides the test fold into an equal ratio of each class.

Clustering

- Link
- In **KMeans**, the K is the centroid that determines the number of groups.
- Steps in KMeans:

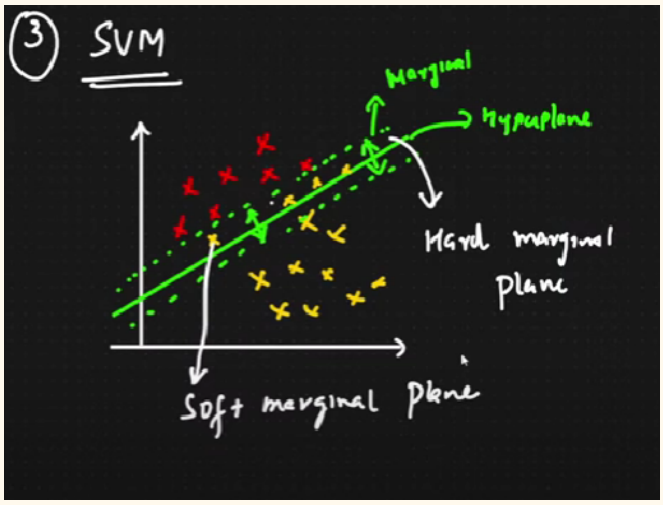
    - Try different k values.
    - Initialize the k number of centroids. Find euclidean distance. Draw a plus sign over the data points to segregate them and assign them according to their distance from the centroids.
    - Take the average of each group. This is done to update the centroid. The centroid will move and then come to the centre of each group. kl
- Elbow method: to understand the K value for the first step, we iterate over numbers and draw a graph between K and WCSS (within-cluster sum of squares).
- To validate the k value found using the elbow method is done by silhouette score.
- If unsure about the number of clusters, outliers in data, computational efficiency, do not use k means.
- **Hierarchical:** combines the points near to each other. From shortest distance to large. Then these groups are combined again according to their distance. As seen in a dendrogram.
- Understand the number of groups in a dendrogram: find the longest vertical line with no horizontal line passing through it.
- **Which process takes more time?** Hierarchical clustering takes more time than KMeans.
- Silhouette score: ranges between -1 till +1. A better model has a value of +1. The model, which has a score near -1, says that the distance between the centroid and the points is far from the cluster.
- **Two-step cluster analysis:**

DBScan Clustering

- Epsilon, min points, core points, border points, and noise points.
- Noise point is neglected and is treated as an outlier. Never taken inside a group.
- Border point is taken inside the group of clusters.
- There is no under or overfitting in clustering. Rather we validate the number of clusters found from the elbow method concerning the silhouette score.

SVM

- Aim: we create the best-fit line, a marginal plane, and a hyperplane. This is done to split the points.
- The distance between the marginal planes should be maximum.
- In the soft marginal plane, we find errors or overlapping points.

- 

-

Neural Networks

- Forward propagation.
- ANN - Artificial NN, CNN - Convolutional, RNN - Recurrent NN.
- ANN for traditional business problems, RNN for text, and CNN for image. RNN is more advanced than the other two.
- Inputs wi*xi go to the processing unit, and the sum of all inputs is calculated, z. Wi is weights that are found using gradient descent. Z = transpose(w)*x. Both are matrices and give a scalar quantity.
- Activation function = g(z) is known as a reactivated linear unit, and if z is greater than 0 or some threshold value, then the neuron fires; else does not fire.
- If g is a sigmoid function, this algorithm becomes logistic regression.
- Cost function is a convex function.
- Average cost function of training data is high, which tells how bad our algorithm is.
- This cost function takes all the input combinations of all biases and weights and gives out one number.
- **Back-propagation** - in this, we feed the loss or the cost back to the neurons such that we can fine-tune the weights based on which the optimisation function, like gradient descent, can help us find weights that will result in a smaller loss or cost in the next iteration.
- **ANN** - learn by example, like pattern recognition or data classification through a learning process.
- The purpose of the activation function is to introduce non-linearity to the output. So that it learns more complex tasks.

- **Activation function -** a **mathematical function** added to a neural network to learn complex patterns. It introduces **nonlinearity** to the output of a neuron. We need this because most real-world data is non-linear, and we need the neurons to understand this non-linearity.
    - **Sigmoid** - takes real-valued input and ranges it between 0 and 1. This activation function is 1/1+e^(-y), where y is the summation of wi*xi + bi.
    - **Tanh -** takes real-valued input and ranges it to between -1 and 1. $f(x) = tanh(x) = \frac{1-e^{-2x}}{1+e^{-2x}}$
    - **ReLU -** rectified linear unit. Replaces negative values with zero. Better convergence than the above two. Has no saturation regions.
    - **Leaky ReLU -** helps solve dying relu situations. This means that the new weight will be equal to the new weight. It helps perform backpropagation when inputs are negative. The gradient is a non-zero value instead of zero avoiding dead neurons.
- Pillow library:
    - Used to convert images to NumPy array and produce an image from NumPy array.
    - Also used for cropping, greying, rotating, transposing an image and many more functions.

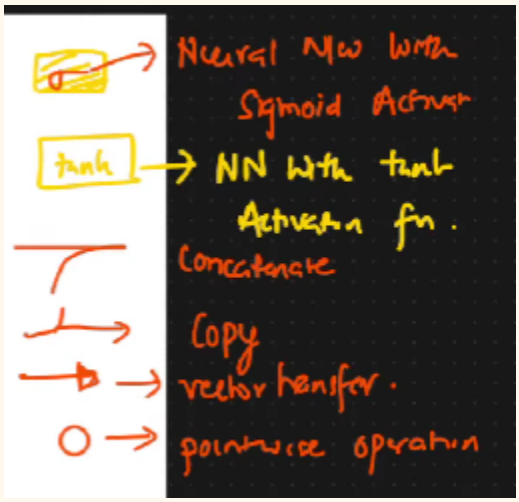
CNN

- 0: Black, 255: White. BW has only one channel.
- 5X5X3 for RGB channel. There are 3 channels for each layer.
- We try to use min-max scaling in all our pixels to bring the value between 0 to 1.
- Using filters, we try to extract information on the image.
- Vertical edge, filter kernel
- When passing a 6x6 through a 3x3 filter we get an output of 4x4. The formula is: n-f+1 (6-3+1).
- Since the output dimension is reducing, we apply padding to retain the lost information. Protecting the image by adding more layers to it.
- We update filter values in backpropagation based on the input image. After the convolution process, we apply the relu activation function to each value. We select this since the derivative can be found during backpropagation.
- Stride: check more. (n+2p-f+1)/s.

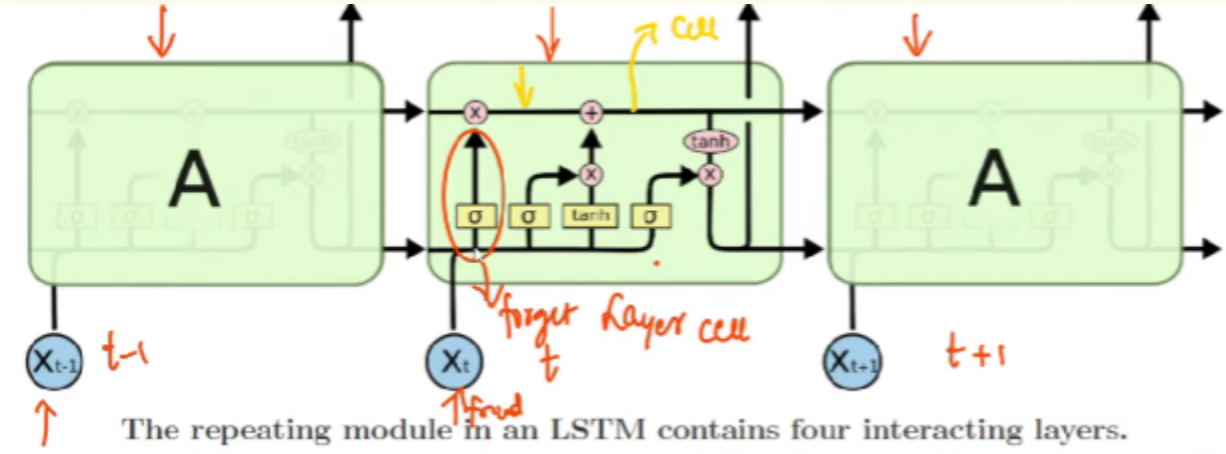
- Image passed to a filter will get an output along with relu. These 3 things together are a convolution operation. These are stacked horizontally. We need to learn and update based on the image inputs.
- Max, min, or avg pooling:
- Flattening layer: the max pooling is elongated. This becomes a part of the ANN dense layer.
- 

-

LSTM RNN

- In LSTM, we are supposed to remember the context as well. Hence it is used most in textual problems. This is done by using an attention mechanism which gives context to each word in the input.
- Attention weights are given to each word according to its importance.
- 

- Memory cell: can add or remove info
- When sentences are long, there is a chance that the first half of the sentence doesn't contain important info. LSTM helps in forgetting this part and only focusing on what is ahead and important.
- Forget layer cell: used for only one critical piece of information.
- Joe likes chicken, but his friend likes shrimp. What does his friend like? Should be able to forget the first half (what joe likes).

- We pass this to the forget layer through the sigmoid function. This sigmoid function is trained to forget what is passed to it, and the output it gives is 0.
- 

The repeating module in an LSTM contains four interacting layers.
- This is for sentences where the context switch happens. In the memory cell, the information doesn't get removed.
- The above image has 3 layers. But the middle one has 3 gates in itself. One for adding, forgetting, and an output gated layer.
- Pre-padding: zeroes added to first to make the sentences of equal length.
- Post-padding: zeroes added to the last to make the sentences of equal length.
-

Important points

- ➢ Low R-square means that the model cannot learn from training data.
- ➢ Variables are insignificant - variables are not contributing to the model.
- ➢ Lasso is called L1 because it minimises the absolute sum of coefficients, whereas the ridge is called L2 because it minimises the squared absolute sum of coefficients.
- ➢ Supervised learning - when the data we have is labelled.
- ➢ Labelled data - for every y, there is an x, and every feature and column is well understood, and the model can learn from it.
- ➢ The two parts of supervised learning are regression and classification.
    - Regression - the target is numeric, for example - linear regression. Performance metrics - RMSE, MAPE, MAE, MSE.
    - Classification - the target is categorical, for example - logistic regression and decision tree. Performance metrics - accuracy, precision, recall, f1 score, auc of roc.
- ➢ Dot products tell us the similarity between two vectors. If it is 0, there is no similarity.
- ➢ Orthogonal - two uncorrelated dimensions.

- If there are missing values in data, then we won't be able to calculate beta hat in linear regression.
- Why do we calculate the mean value and others? Because the mean is a sufficient statistic to give information on parameters.
- KNN doesn't have TPR or FPR because KNN has many classes, not just positive and negative. However, there is a confusion matrix. It is a multinomial classification.
- We look for homogeneity while doing classification models.
- Spearman's rank correlation is used to capture nonlinear and linear properties that are not done with ordinary correlation (Pearson) practices.
- P-value is a probability that if p is less than or equal to 0.05, there is a 5% probability that the null hypothesis is correct. Hence we reject the null hypothesis when p is less than or equal to the significance value.
- Train vs test vs validation. Validation data split is done to hyper-tune the model. Like when we do cross-validation like gridsearchcv and more. If there are 1000 data points, 200 are for validation, and 800 are for training data. This follows for n number of iterations.
- **Why random forest instead of decision tree?** A decision tree usually has low bias and high variance, and a random forest has low bias and variance.
- How can you convert Pareto to the normal distribution? Box cox transformation.
- If there are two graphs, one right-skewed and one left, what is the relation between mean, median, and mode? Left: mean>median>mode and mode>median>mean.
- Difference between fit_transform and transform. Why is it only done to train set and transform to test set? To avoid data leakage.
- Difference between normalisation and standardisation.
- **Bagging:** several independent models are fit together, and their predictions are averaged. The main aim is to reduce the variance.
- **Boosting**: it is the sequential set of all the models combined, and these models are initially weak learners, and when put together, they become strong learners to give good results. Bagging merges the same type of prediction models and boosting different.
- **Do you normalise or standardise data points in RF or KNN?** We don't need to normalise in RF because we do splits of the features. If we minimise the data, the feature won't be important anymore. But in KNN, yes, because we do standardise because we compute manhattan or euclidean distances.
- **Is RF impacted by outliers?** No. check more.
- **For which algorithms are feature scaling required?** ANN, LR, LogR, KNN, Kmeans. Algorithms related to distance-based, large calculations are there so required. Then wherever gradient descent is where it is required.
- White and black box model.

- **Dropout** is used in a neural network to avoid overfitting. It is a kind of regularisation in neural networks. Some nodes are not sent into the network in either forward or backward passes during the training stage.
- If we have an outlier in a normal distribution, we use IQR to treat it.
- But in skewed, we have to see the area for outliers and not use IQR.
- If the data is normally distributed, calculate the upper and lower boundary: mean +- 3*stdev, else, using IQR, find the 25 and 75 percentile and then lower and upper bridge. If the data is skewed and not normal, then instead of multiplying IQR by 1.5, do it by 3.
    - Q1 = np.percentile(column, 25, interpolation = 'midpoint')
    - Q3 = np.percentile(column, 75, interpolation = 'midpoint')
    - IQR = Q3-Q1
    - column.quantile(0.25) - (IQR*1.5(or 3))
    - column.quantile(0.75) + (IQR*1.5(or 3))
- Mean imputation is not always a good choice because it doesn't take into account feature correlation and also reduces variance and increases the bias.
- Role of hypothesis in linear regression: it is used to test the significance of independent variables in the regression model.

URLs

- What's a good value for R-squared? (duke.edu)
- http://rishy.github.io/ml/2015/07/28/l1-vs-l2-loss/
- 5 Regression Loss Functions All Machine Learners Should Know | by Prince Grover | Heartbeat (fritz.ai)
- Top 75 Statistics Interview Questions & Answers 2022 - Intellipaat
- Post | Feed | LinkedIn

Python

```
Pd.options.display.max_rows = True
Pd.options.display.max_columns = True
df.isnull().sum()

from sklearn.preprocessing import StandardScaler, MI
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error
From sklearn.model_selection import train_test_split, GridSearchCV
```

From sklearn.tree import DecisionTreeClassifier
Find p-value: import scipy.stats; scipy.stats.norm.sf(abs(xx))

Miscellaneous

- Pareto analysis: used to find the most important areas, factors or variables when there are multiple. It follows the 80/20 rule. 80% of the consequences come from the 20% of causes.
-