**# Few Important Points -**

1. An “algorithm” in machine learning is a procedure that is run on data to create a machine learning “model.” For example, we have algorithms for classification, such as k-nearest neighbors. We have algorithms for regression, such as linear regression, and we have algorithms for clustering, such as k-means. scikit-learn library provides implementation of many algorithms. While a “model” in machine learning is the output of a machine learning algorithm run on data. The model is the “thing” that is saved after running a machine learning algorithm on training data
2. Keras is a deep learning library used to develop and evaluate deep learning models. It is tightly integrated with Tensorflow. Sklearn is a library for general machine learning and it provides utilities that are helpful in deep learning models.
3. Machine learning learns from historical data and provides statistical tools for exploring data (Supervised, Unsupervised, Reinforcement). Deep learning based on multi neural network architectures (ANN - numeric, CNN - images, RNN - time series). Being it ML or DL, it derives an AI application. Data Scientist works on AI, ML DL using mathematical tools like probability, Linear Algebra, Statistics
4. Dimensionality reduction methods create new features from the data, feature selection chooses existing features from the data.
5. Your results may vary given the stochastic nature of the algorithm or evaluation procedure, or differences in numerical precision. Consider running the example a few times and compare the average outcome.
6. The most common approach to dimensionality reduction is called principal components analysis or PCA.
7. Bining and K-bins-Discretization is used to convert numeric data into binary or into multiple categories based on the threshold. Helpful in classification problems
8. Similar to image data augmentation, we can create test-time augmentation for our normal numeric/categorical data for regression and classification problems. Given a row, it adds some gaussian noise and generates a new row
9. A machine learning pipeline is used to help automate machine learning workflows. Pipeline has the steps array which takes the data transformers like (normalizer, standarizer etc) and the model like (LinearRegression, RandomForest etc) and then we can fit and predict results with the pipeline object
10. Everything is converted into numbers before training
11. Variance refers to the spread of a data set around its mean value, while a covariance refers to the measure of the directional relationship between two random variables.

**# Best Practices**

1. If performance of model doesn’t improve, try using different random seed, reduce learning rate, add more layers, add more data etc
2. Rather than picking your favorite algorithm, try 10 or 20 algorithms. Double down on those that show signs of being better in performance, robustness, speed or whatever concerns interest you most. Rather than picking the common parameters, grid search tens, hundreds or thousands of combinations of parameters.
3. Play with tensorflow playground with different hyper parameters and use its reference in the code
4. Create reusable script to automatically try 10, 20, 50 algorithms across a variety of libraries and implementations on a small dataset
5. You may save and load models to resume the training next day

**# Regression**

1. Loss function for Regression problems. MAE - 1/m \* Summation of |ai - yi)|, MSE - 1/2m \* Summation of square of |ai - yi|. ai is predicted value while yi is actual value
2. For a regression problem, better transform the target variable as well. Automatic transformation can be done with TransformedTargetRegressor. It does the inverse transform at the time of prediction to get correct results. We can also apply transformation manually but it won't be possible to use cross\_val\_score for different sets in case of k-fold splitting
3. 100-MAPE(Mean Absolute Percentage Error) or 100-MedianAPE(in case of outliers) are ways to measure accuracy of regression models. These are event level/row level accuracy.
4. Also check for overall accuracy with 100-bias. Bias is the percentage of difference between sum of predicted values and sum of actual values. Never consider only bias as the measure to find accuracy.

**# Classification**

1. MultiLabel Classification - Sample can belong to multiple categories. Say an image has cat and dog together. It belongs to both categories. Say a movie is action as well as comedy. So final layer activation func is Sigmoid (1,0,0,1) for (action, romance, drama, comedy) and loss has to be binary\_crossentropy
2. Loss function for Binary Classification. Binary Cross Entropy - -1/m \* Summation of [yi\*log(ai) + (1-yi)\*log(1-ai)]
3. Loss function for Multiclass classification. Categorical Cross Entropy - 1/m \* Summation of [y11\*log(a11) + y12\*log(a12) + y13\*log(a13) +y14\*log(a14) + ....]. Say yi = [0,1,0,0] and ai = [0.1,0.5,0.3,0.1]
4. For classification problems, use LabelEncoder to transform the target variable.
5. We may train multiple models (Random Forest, XgBoost, Logistic Regression etc) and then predict the category. We may take a vote from multiple models and predict the appropriate category. This ensemble learning technique can also be applied to regression.

**# Neural Network**

1. We can even have a neural network model for combined regression and classification - <https://machinelearningmastery.com/neural-network-models-for-combined-classification-and-regression/>
2. Relu activation function and he\_normal kernel\_initializer together are a good practice. Use ReLU with MLPs, CNNs, but Probably Not RNNs (like LSTM). When using ReLU in your network, consider setting the bias to a small value, such as 0.1. Also, use the L1 penalty
3. Activation Functions - Linear takes as it is (x), Sigmoid converts the value b/w 0 to 1 ( 1/1+exp(-x)) , tanh b/w -1 to 1, relu converts all negative to 0 so, max(0, value), LeakyRelu takes some negative values, ELU (Exponential Linear Unit) have negative values which pushes the mean of the activations closer to zero, PReLU (Parametric ReLU) learns parameters that control the shape and leaky-ness of the function.

**# Missing Values**

1. Marking missing values with a NaN (not a number) value in a loaded dataset using Python is a best practice. Then replace this NaN with some mean, median, interpolate, mode etc. Also, try adding an extra column in the dataset which will have the value as 1 or 0 depending on if any value in row has NaN (by summing all). This can be useful to model. Also, try with sklearn's SimpleImputer which effectively adds one additional column for each input variable that contains missing values and may offer benefit to the model.
2. While reading csv in dataframe, we can pass a parameter na\_values=?. It will replace all ? values with NaN.
3. Try IterativeImputer to fill in all missing values in all columns - <https://machinelearningmastery.com/iterative-imputation-for-missing-values-in-machine-learning/>

**# Feature Scaling**

1. Try only normalizing non-Gaussian like input variables, Try only standardizing Gaussian like input variables, Try with both and see what gives the best results.
2. Transform data to become gaussian and then apply standardization
3. After normalizing with normalization, we can use PowerTransformer/QuantileTransformer etc to make the data more Gaussian to see if performance improves.
4. Normalization preserves the distribution but scales down values b/w 0 & 1 (MinMaxScalar). While Standardization reduces the effect of outliers.
5. RobustScalar takes care of outliers as its formula is x-median/iqr75-iqr25. We can even change inter quartile ranges
6. If a dataset has mixed types, ColumnTransformer allows you to selectively apply data transforms to different columns in your dataset.

**# Feature Selection**

1. Use sklearn SelectKBest for feature selection for numerical data. It will give the score of each feature, the higher the value the more important it is. Using ANOVA f-test which uses sklearn GridSearchCV, we can also get the best number of selected features to be given to model
2. For feature selection on categorical data, use Chi-Squared feature selection. With this score\_func=chi2 is passed as argument to SelectKBest. Similarly, we have another technique i.e. score\_func=mutual\_info\_classif. So, try model with all features and selected features with chi2 and mutual\_info\_classif to see which performs the best
3. Using Recursive Feature Elimination (RFE) to select the optimal features and giving only those to model. (<https://machinelearningmastery.com/rfe-feature-selection-in-python/>)

**# Splitting Data**

1. Divide the dataset into 3 parts. Divide the training, validation and test sets in such a way that their distribution is similar. In case distribution is different (say images from web and camera). Read up to split properly - <https://www.freecodecamp.org/news/what-to-do-when-your-training-and-testing-data-come-from-different-distributions-d89674c6ecd8/>
2. K-fold cross validation helps to avoid overfitting
3. It is better to use RepeatedStratifiedKFoldClassifier than normal train-test split. K-fold takes different samples in test data and finally takes a mean to show the accuracy score. Stratified k-fold makes sure the data of each category is present in train and test sets
4. Data preprocessing must be fit on the training dataset only i.e. Split should be done before preprocessing
5. The test set should ideally not be preprocessed with the training data. This will ensure no 'peeking ahead'. Train data should be preprocessed separately and once the model is created we can apply the same preprocessing parameters used for the train set, onto the test set as though the test set didn't exist before.

**# Outliers**

1. Common algorithms to automatically detect outliers in the dataset are IsolationForest, LocalOutlierFactor etc. Always try with manual process and these algorithms to check what gives the best results
2. If there are any values which are less than 1.5 times of first quartile or greater than 1.5 times of third quartile, they can be considered as outliers

**# Encoding Categorical Data**

1. If categorical features are ordinal i.e. they have some rank (10, 12, Graduation, PostGraduation etc), use LabelEncoding with proper mapping (0,1,2,3)
2. When the number of categorical features present in the dataset is less, use one-hot encoding
3. Consider using Binary encoding when no of categories are very large and there is no ordinality
4. When no of categories are very large, apply one-hot encoding on categories that make 90% of data, for the remaining ones, create a column called others
5. If no of categories are large and data is almost equally distributed, use Binary Encoding. It can categorize 16 different types into 4 columns (0000-1111) and so on.
6. Unlike LabelEncoder, OrdinalEncoder assigns values as 0,1,2. Use OrdinalEncoder where order is important
7. LabelEncoder, by design, has to be used on the target variable and not on feature variables
8. We can use a one-hot encoding up to thousands and tens of thousands of categories. Also, having large vectors as input sounds intimidating, but the models can generally handle it.

**# Hyperparameter and other aspects -**

1. Improve Bias/variance (Model to have low bias and low variance)
2. If the validation set error is much more than the train set error, the model is overfitting and has a high variance
3. When both train and validation set errors are high, the model is underfitting and has a high bias
4. If the train set error is high and the validation set error is even worse, the model has both high bias and high variance
5. And when both the train and validation set errors are small, the model fits the data reasonably and has low bias and low variance
6. Basically high bias results in underfitting and high variance results in overfitting
7. To reduce bias, try bigger networks, train models for longer periods, try different neural network architectures.
8. To reduce variance, get more data, use regularization, or try different neural network architectures.
9. Regularizing Neural Network -
10. If we can't add more data to reduce variance, we can add regularization to the cost function. ƛ is the regularization parameter which we can tune while training the model. Adding ƛ results in more reduction in weight and so L2 is also called weight decay. Adding high value of ƛ will result in very lower weights making the network simpler and thus reduce overfitting
11. Dropout is another regularization technique to reduce overfitting. Suppose we add a dropout of 0.5 to all images. The model will randomly remove 50% of the units from each layer and we finally end up with a much simpler network.
12. Use Data Augmentation to increase data
13. Use Early stopping to stop the training at the point where the validation set error starts to increase.
14. Normalization helps the algorithm to run faster as it makes the cost function symmetric which helps gradient descent to find the global minima quickly. Use the same normalization on train and test sets
15. Large weights make the gradients very large and the learning process slow. Similarly, small weights make the gradients very small, and as a result, learning will be slow. Initializing the weights randomly helps to break symmetry. We want to make sure that different hidden units learn different patterns.
16. Gradient checking is used to find bugs (if any) in the implementation of backpropagation.
17. For a large training data, say 50000000, If we process through all of these training examples in every training iteration, the gradient descent update will take a lot of time. So, use mini-batch gradient descent. If mini-batch size is m, it becomes batch gradient descent and it takes long time per iteration, if size is 1, it becomes stochastic gradient descent, here in every iteration we are taking just a single example, it can become extremely noisy and takes much more time to reach the global minima. The size of the mini-batch should not be too large or too small.
18. If the training set is small, we can choose a mini-batch size of m less than 2000
19. For a larger training set, typical mini-batch sizes are: 64, 128, 256, 512
20. The ultimate goal is to reach the global minima. Large learning rate could result in large upgrade steps and we might not reach global minima and a lower learning rate makes gradient descent slower. We want a slower learning in the vertical direction and a faster learning in the horizontal direction which will help us to reach the global minima much faster. This can be achieved using gradient descent with momentum where we have a parameter beta that acts as fiction.
21. There is another optimizer called RMSProp which restricts the oscillations in the vertical direction
22. Adam is essentially a combination of momentum and RMSprop. Adam helps to train a neural network model much more quickly than the techniques we have seen earlier. There are a range of hyperparameters used in Adam and some of the common ones are:
23. Learning rate α: needs to be tuned
24. Momentum term β1: common choice is 0.9
25. RMSprop term β2: common choice is 0.999
26. ε: 10^ -8
27. Learning rate decay is a technique for training modern neural networks. It starts training the network with a large learning rate and then slowly reducing/decaying it until local minima is obtained
28. Few common hyperparameters we frequently work with in a deep neural network:
29. Learning rate – α
30. Momentum – β
31. Adam’s hyperparameter – β1, β2, ε
32. Number of hidden layers
33. Number of hidden units for different layers
34. Learning rate decay
35. Mini-batch size
36. Learning rate usually proves to be the most important among the above. This is followed by the number of hidden units, momentum, mini-batch size, the number of hidden layers, and then the learning rate decay.
37. Normalize the activations of the hidden layer(s) so that the weights of the next layer can be updated faster.
38. Vanilla linear regression does not have any hyperparameters. Variants of linear regression (ridge and lasso) have regularization as a hyperparameter. The decision tree has max depth and min number of observations in the leaf as hyperparameters.
39. We can use GridSearch, RandomSearch, smart hyper-parameters like Spearmint (hyperparameter optimization using Gaussian processes) and Hyperopt (hyperparameter optimization using Tree-based estimators).
40. Gini Index and Entropy tells how pure is the split in the Decision tree. Lower the value, the better it is. It takes the feature and the condition say salary > 50000 and performs the split. It takes all permutation and combination of different features and conditions and choose the one with minimum gini index or Entropy
41. Use Bagging and Boosting techniques for better performance.
42. Multicollinearity means the two independent variables are highly corelated. These will affect the coefficients in regression and so the performance. So, check the correlation b/w all variables. If there is high correlation (say >0.8) b/w 2 variables, remove one which has weaker strength with dependent variables. For larger no. of columns use Lasso/Ridge regression to remove multicollinearity

**# Time Series**

1. We can convert time series data to supervised learning data by taking prev timestamp as input variable and next time stamp as output variable and then apply regression algo like Xgboost, RandomForest etc
2. shift() in Pandas can be used to create new framing of the time series problem. Try with different framing for better results
3. We can’t use k-fold cross validation, instead walk forward validation is used where the dataset is first split into train and test sets by selecting a cut point, e.g. all data except the last 12 months is used for training and the last 12 months is used for testing. It respects the temporal ordering of observations
4. Prophet is used for univariate time series forecast
5. ARIMA is used for univariate time series forecasts. It can handle trend but not the seasonal component
6. SARIMA is used for univariate time series forecasts. It can handle trends as well as seasonal components. Implementation is rather called SARIMAX where X means it supports exogenous variables (these are unaffected by other variables)
7. If mean and variance don’t change over time, data is stationary. E.g - excess stock returns always fluctuate around zero.
8. Cross-validation is a method that does a repeated train test evaluation. If you use 3-fold cross-validation, you will split your data set into three equal parts. You will then fit three times the same model on two-thirds of the data set and use the other third for evaluation and you can use the average as the final metric. Disadvantage is that each of the models does not use the same number of years in the training data. An alternative is to do a rolling split (always 5 years train, 5 years test), but here the disadvantage is that you can never use more than 5 years for training data.
9. Random Forest will always predict value within minimum and maximum value in training data. If there is an increasing or decreasing trend in the data, Random Forest Regressor will not be able to formulate.
10. Exponential smoothers are good for short term forecasting especially on rapidly changing time series. It is used to forecast the immediate future.
11. Index and parse Date column at the time of reading csv and tell the data frame of what is the frequency like Daily, weekly etc.
12. df[‘date\_column’][‘2021-01-01’:].plot.figsize(9,6) will plot the data from Jan 2021
13. Holt's winter method should only be used if we have trend and seasonality
14. Additive trend - If rate of change is constant.
15. Multiplicative trend - If rate of change is changing exponentially.(not constant)
16. Additive Seasonality - Seasonal magnitude is same over time (similar sales in Diwali throughout years)
17. Multiplicative Seasonality - Seasonal magnitude is different over time (dissimilar sales in Diwali throughout years)
18. Use correct additive or multiplicative trend or seasonality based on the input data
19. It is different from regression. In regression, we predict based on different variables. In time series, we predict based on the same variable but at previous times. Regression is interpolation as error will have some confidence range and will be almost the same for all points. TimeSeries is extrapolation where error for next day will be less and will keep increasing for future dates
20. ACF - Price at t depends directly on Pt-3 as well as Pt-3 -> Pt-2 -> Pt-1 while for PACF - Price at t depends directly on Pt-3 and not the subsequent ones.
21. For Pt = a0 + a1Pt-1 + a2Pt-2 + a3Pt-3 + a4Pt-4 + e. If we see high correlation (positive/negative) only for t-2 and t-4, best model will be P = a0 + a2Pt-2 + a4Pt-4 + e
22. Based on diminishing/decaying ACF graphs, we are likely dealing with AR processes. PACF graph shows, if we should start with an AR model with lags say 1,3,6,7,10
23. Differencing is a popular technique to make your data stationary
24. ACF and PACF may not tell much for problems like stock price prediction. If you clear lags, it may be useful
25. For Eq - Pt = a1Pt-1 + e. If a1 is less than 1, data is stationary. If a1>1, data is not stationary. If a1=1, it means it has unit roots and data may or may not look stationary.
26. A stationary time series is one whose statistical properties such as mean, variance, autocorrelation, etc. are all constant over time.
27. Dickey fuller and Augmented Dickey fuller tests are some ways to find if data is stationary. p value should be less than 0.5 to confirm stationarity
28. Time series has white noise if mean is 0, std dev is constant and correlation b/w lags is 0 (i.e. no correlation). Such time series are not predictable. We can test for white noise with visual checks, global vs local checks (check mean etc for whole data and with some time windows), Check ACF graph having 0 or very small lags.
29. Date column should be parsed to datetime object from string and should also be indexed
30. Residuals are prediction v/s test data. We should plot the residuals
31. Find MAPE to see how far we are from actual data
32. Based on ACF, PACF graphs, you can put the lags for AR model
33. when looking into the model summary, it shows p value for different lags. only those lags where p value is less than 0.5 should be considered. Similarly if constant is greater than 0.5, don't keep the constant
34. Splitting data into 70-30 may give good results for some dates that are close to training data, but results will get worse for the dates which are away from training data. Ideally, we should evaluate multiple models like evaluate k-3 based on 1,2,...k-4 then evaluate k-2 based on 1,2,....k-3 and so on. This is known as rolling forecast
35. MA(1) is represented as mean + a0et-1, E.g take mean as 10 and a0=0.5 and keep evaluating MA based on error in previous lags. It always moves around mean.
36. ARMA is a combination of AR and MA. ARMA(1,1) = (a0 + a1Pt-1) + (b1et-1)
37. Anything outside the band in ACF tells the order of MA bits while anything outside the band in PACF tells the order of AR bits
38. ARIMA is used when there is an obvious linear upward or downward trend. I is adding differencing (1st, 2nd etc whichever makes the data stationary)
39. Seasonality is a predicted repeating pattern. To make data stationary, we need to remove seasonality. For removing yearly seasonality, we can simply do - Zt = Yt+365 - Yt. Cycles looks like seasonality but these are trends that happen over the course of say several years (2 yrs, 3 yrs, 5 yrs etc) and are not predictable.
40. SARIMA - (p,d,q)(P,D,Q)m where m is no. of times we see seasonality in a year. d says backshift my time series while D says backshift my time series by m time periods. If there is pure seasonal behavior in AR and MA, put p,q as 0 and P,Q as 1. d can be 1 to make data stationary and the D as 0
41. Rolling Forecast helps to achieve better results i.e instead of predicting say 6 months together. Predict for 1 month and then add this in training data and predict for 2nd month and so on.
42. ARCH - First fit your best model and then plot your residuals. If you see different patches of volatility (for some time it predicts well, for sometime it doesn't), good to go for ARCH (AR Conditional Heteroskedasticity). ARCH(1) tells if my model is very volatile today, it will also be volatile tomorrow.
43. In case there is a burst like it stays around its average for a few days and then a sudden spike and then comes back to around its average for a few days and then a sudden spike, we should use GARCH (Generalized ARCH). It takes into account volatility as well as value of the previous day. Library required - from arch import arch\_model. Good to use to predict the volatility of stock market
44. ARMA(1,1) - If we know the value of yesterday (AR), the error of yesterday (MA) and the error of today (e), we can tell the value of today. Usually we don't know the error of today (a0Pt-1 + b0et-1 + e)
45. If we care about multiple variables, say the sale of cars and bikes, use the VAR model (Vector AR). There can be 4 factors (car sales of previous day affect car sales today, bike sales of previous day affect bike sales today, car sales of previous day affect bike sales today, bike sales of previous day affect car sales today). So predict car prices with not just previous car prices but also previous bike prices. Fit the model with max\_lags based on PACF graph and then find the appropriate lags for bike and car for predicting car price based on the model summary and find the appropriate equation for prediction like -
46. Eq for Price of car = a0Ct-1 + a1Ct-2 + a3Bt-5. a0, a1, a2 are coefficients and Ct-1, Ct-2, Bt-5 are the smallest probabilities that we get in the model summary. Another e.g of VAR can be predicting prices of houses (say in Baner) based on previous prices of Baner as well as previous prices of Aundh as Baner and Aundh are nearby.
47. Cleaning time series may involve normalizing data(to make in range), exponentiate(to bring it as linear growth), differencing, removing volatility, removing seasonality and finally applying Dickeyfuller to check for stationarity and then predict. At last undo all transformations one by one to see actual data (code mathematics applied)
48. To detect anomalies, take the deviations by month to see if there is any spike in variance for a particular month. Say we see the spike in variance in Dec month. To remove this anomaly, take the mean of Dec month of previous years and replace the value where we had anomaly. Also there are robust methods like STL for anomaly detection
49. If the time series is not stationary, we can study its behavior only for that time period. Each period of the time series will have its own distinct behavior and it is not possible to predict or generalize for future time periods if the series is not stationary.

**# Clustering**

1. Grouping an unlabelled data is clustering
2. The large value of k in K Means increases the performance, but also increases the risk of overfitting

**# CheatSheets**





