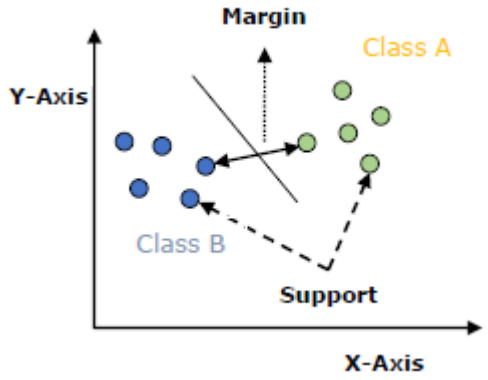
MACHINE LEARNING – WORKSHEET 3

Q1 to Q15 are subjective answer type questions, Answer them briefly.

1. Give short description each of Linear, RBF, Polynomial kernels used in SVM.

Answer: Support vector machines (SVMs) are a set of supervised learning methods used for classification, regression and outlier's detection. An SVM model is basically a representation of different classes in a hyperplane in multidimensional space. The hyperplane will be generated in an iterative manner by SVM so that the error can be minimized. The goal of SVM is to divide the datasets into classes to find a maximum marginal hyperplane (MMH).



The followings are important concepts in SVM −

- **Support Vectors** − Datapoints that are closest to the hyperplane is called support vectors. Separating line will be defined with the help of these data points.
- **Hyperplane** − As we can see in the above diagram, it is a decision plane or space which is divided between a set of objects having different classes.
- **Margin** − It may be defined as the gap between two lines on the closet data points of different classes. It can be calculated as the perpendicular distance from the line to the support vectors. Large margin is considered as a good margin and small margin is considered as a bad margin.

The main goal of SVM is to divide the datasets into classes to find a maximum marginal hyperplane (MMH) and it can be done in the following two steps −

- First, SVM will generate hyperplanes iteratively that segregates the classes in best way.
- Then, it will choose the hyperplane that separates the classes correctly.

SVM Kernels:

SVM algorithm is implemented with kernel that transforms an input data space into the required form. SVM uses a technique called the kernel trick in which kernel takes a low dimensional input space and transforms it into a higher dimensional space. In simple words, kernel converts non-separable problems into separable problems by adding more dimensions to it. It makes SVM more powerful, flexible and accurate. The following are some of the types of kernels used by SVM.

Linear Kernel

It can be used as a dot product between any two observations. The formula of linear kernel is as below −

K(x,xi)=sum(x∗xi)K(x,xi)=sum(x∗xi)

From the above formula, we can see that the product between two vectors say x & xi is the sum of the multiplication of each pair of input values.

Polynomial Kernel

It is more generalized form of linear kernel and distinguish curved or nonlinear input space. Following is the formula for polynomial kernel −

k(X,Xi)=1+sum(X∗Xi)^dk(X,Xi)=1+sum(X∗Xi)^d

Here d is the degree of polynomial, which we need to specify manually in the learning algorithm.

Radial Basis Function (RBF) Kernel

RBF kernel, mostly used in SVM classification, maps input space in indefinite dimensional space. Following formula explains it mathematically −

K(x,xi)=exp(−gamma∗sum(x−xi^2))K(x,xi)=exp(−gamma∗sum(x−xi^2))

Here, *gamma* ranges from 0 to 1. We need to manually specify it in the learning algorithm. A good default value of *gamma* is 0.1.

2. R-squared or Residual Sum of Squares (RSS) which one of these two is a better measure of goodness of fit of

model in regression and why??

Answer: R-squared is a goodness-of-fit measure for linear regression models. This statistic indicates the percentage of the variance in the dependent variable that the independent variables explain collectively. R-squared measures the strength of the relationship between your model and the dependent variable on a convenient 0 – 100% scale.

$$R^2 = \frac{\text{Variance explained by the model}}{\text{Total variance}}$$

R-squared is always between 0 and 100%:

- 0% represents a model that does not explain any of the variation in the response variable around its mean. The mean of the dependent variable predicts the dependent variable as well as the regression model.
- 100% represents a model that explains all of the variation in the response variable around its mean.

R-squared has Limitations:

- we cannot use R-squared to determine whether the coefficient estimates and predictions are biased, which is why you must assess the residual plots.
- R-squared does not indicate if a regression model provides an adequate fit to your data. A good model can have a low R^2 value. On the other hand, a biased model can have a high R^2 value!

Residual sum of squares is used to help you decide if a statistical model is a good fit for your data. It measures the overall difference between your data and the values predicted by your estimation model (a "residual" is a measure of the distance from a data point to a regression line). Total SS is related to the total sum and explained sum with the following formula:

Total SS = Explained SS + Residual Sum of Squares.

the residual sum of squares (RSS), also known as the sum of squared residuals (SSR) or the sum of squared errors of prediction (SSE), is the sum of the squares of residuals (deviations of predicted from actual empirical values of data). Residual Sum of Squares (RSS) is defined and given by the following function:

$$RSS=\sum_{ni=0}(\epsilon_i)^2=\sum_{ni=0}(y_i-(\alpha+\beta x_i))^2$$

Where −

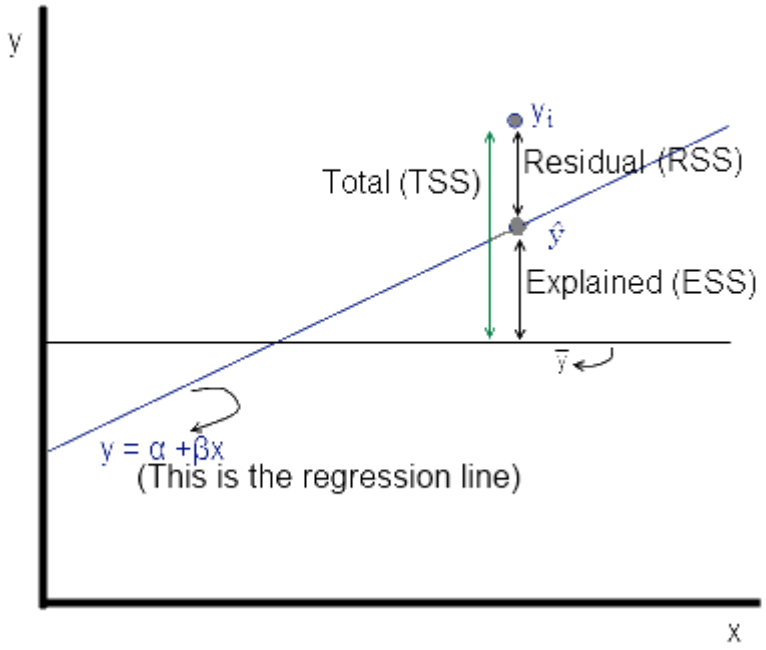
- X, Y = set of values.
- α, β = constant of values.
- n = set value of count

for hard to predict Y variables, smaller values may be "good". Overall, R2 provides a useful measure of how well a model fits, in terms of (squared) distance from points to the best fitting line. However, as one adds more regression coefficients, R2 never goes down, even if the additional X variable is not useful.

3. What are TSS (Total Sum of Squares), ESS (Explained Sum of Squares) and RSS (Residual Sum of Squares)

in regression. Also mention the equation relating these three metrics with each other.

Answer: <u>TSS, RSS and ESS:</u> Consider the diagram below. Yi is the actual observed value of the dependent variable, y-hat is the value of the dependent variable according to the regression line, as predicted by our regression model. What we want to get is a feel for is the variability of actual y around the regression line, i.e., the volatility of ϵ. This is given by the distance yi minus y-hat. Represented in the figure below as RSS. The figure below also shows TSS and ESS – spend a few minutes looking at what TSS, RSS and ESS represent.



$\hat{y}$ is the predicted value of y given x, using the equation $y = \alpha + \beta x$.

y_i is the actual observed value of y

$\bar{y}$ is the mean of y.

The distances that RSS, ESS and TSS represent are shown in the diagram to the left - but remember that the actual calculations are squares of these distances.

$$TSS = \Sigma (y_i - \bar{y})^2$$

$$RSS = \Sigma (y_i - \hat{y})^2$$

$$ESS = \Sigma (\hat{y} - \bar{y})^2$$

Now ϵ = ***observed – expected value of y***

Thus, $\epsilon = y_i - y\hat{}$. The sum of ϵ is expected to be zero. So we look at the sum of squares:

The value of interest to us is $= \Sigma (y_i - y\hat{})^2$. Since this value will change as the number of observations change, we divide by '***n***' to get a 'per observation' number. (Since this is a square, we take the root to get a more intuitive number, ie the RMS error explained a little while earlier. Effectively, RMS gives us the standard deviation of the variation of the actual values of ***y*** when compared to the observed values.)

If ***s*** is the **standard error of the regression**, then

s = sqrt (RSS/(n – 2))

(where n is the number of observations, and we subtract 2 from this to take away 2 degrees of freedom*.)

Now $TSS = \Sigma (y_i - \bar{y})^2$

$RSS = \Sigma (y_i - \hat{y})^2$

$ESS = \Sigma (\hat{y} - \bar{y})^2$

$TSS = ESS + RSS$

4. What is Gini –impurity index?

Answer: Gini index or **Gini impurity** measures the degree or probability of a particular variable being wrongly classified when it is randomly chosen. But what is actually meant by 'impurity'? If all the elements belong to a single class, then it can be called pure. The degree of Gini index varies between 0 and 1, where 0 denotes that all elements belong to a certain class or if there exists only one class, and 1 denotes that the elements are randomly distributed across various classes. A Gini Index of 0.5 denotes equally distributed elements into some classes.

Formula for Gini Index

$$\text{Gini} = 1 - \sum_{i=1}^{n} (p_i)^2$$

where p_i is the probability of an object being classified to a particular class.

While building the decision tree, we would prefer choosing the attribute/feature with the least Gini index as the root node.

5. Are unregularized decision-trees prone to overfitting? If yes, why?

Answer: Yes, because Over-fitting is the phenomenon in which the learning system tightly fits the given training data so much that it would be inaccurate in predicting the outcomes of the untrained data.

In decision trees, over-fitting occurs when the tree is designed so as to perfectly fit all samples in the training data set. Thus, it ends up with branches with strict rules of sparse data. Thus, this effects the accuracy when predicting samples that are not part of the training set.

One of the methods used to address over-fitting in decision tree is called pruning which is done after the initial training is complete. In pruning, you trim off the branches of the tree, i.e., remove the decision nodes starting from the leaf node such that the overall accuracy is not disturbed. This is done by segregating the actual training set into two sets: training data set, D and validation data set, V. Prepare the decision tree using the segregated training data set, D. Then continue trimming the tree accordingly to optimize the accuracy of the validation data set, V.

6. What is an ensemble technique in machine learning?

Answer: Ensemble methods is a machine learning technique that combines several base models in order to produce one optimal predictive model. The goal of ensemble methods is to combine the predictions of several base estimators built with a given learning algorithm in order to improve generalizability / robustness over a single estimator.

Two families of ensemble methods are usually distinguished:

- In **averaging methods**, the driving principle is to build several estimators independently and then to average their predictions. On average, the combined estimator is usually better than any of the single base estimator because its variance is reduced.

Examples: Bagging methods, Forests of randomized trees, …

- By contrast, in **boosting methods**, base estimators are built sequentially and one tries to reduce the bias of the combined estimator. The motivation is to combine several weak models to produce a powerful ensemble.

Examples: AdaBoost, Gradient Tree Boosting, …

7. What is the difference between Bagging and Boosting techniques?

Answer: Bagging and Boosting decrease the variance of your single estimate as they combine several estimates from different models. So, the result may be a model with **higher stability**.
If the problem is that the single model gets a very low performance, Bagging will rarely get a **better bias**. However, Boosting could generate a combined model with lower errors as it optimises the advantages and reduces pitfalls of the single model.

- Both are ensemble methods to get N learners from 1 learner, but, while they are built independently for Bagging, Boosting tries to add new models that do well where previous models fail.
- Both generate several training data sets by random sampling, but only Boosting determines weights for the data to tip the scales in favor of the most difficult cases.
- Both make the final decision by averaging the N learners (or taking the majority of them), but it is an equally weighted average for Bagging and a weighted average for Boosting, more weight to those with better performance on training data.
- Both are good at reducing variance and provide higher stability, but only Boosting tries to reduce bias. On the other hand, Bagging may solve the over-fitting problem, while Boosting can increase it.

8. what is out-of-bag error in random forests?

Answer: Out of bag (OOB) score is a way of validating the Random forest model. The RandomForestClassifier is trained using *bootstrap aggregation*, where each new tree is fit from a bootstrap sample of the training observations zi=(xi,yi). The *out-of-bag* (OOB) error is the average error for each zi calculated using predictions from the trees that do not contain zi in their respective bootstrap sample. This allows the RandomForestClassifier to be fit and validated whilst being trained. The OOB score is computed as the number of correctly predicted rows from the out of bag sample.

9. What is K-fold cross-validation?

Answer: Cross-validation is a resampling procedure used to evaluate machine learning models on a limited data sample. The procedure has a single parameter called k that refers to the number of groups that a given data sample is to be split into. As such, the procedure is often called **k-fold cross-validation**. When a specific value for k is chosen, it may be used in place of k in the reference to the model, such as k=10 becoming 10-fold cross-validation. Cross-validation is primarily used in applied machine learning to estimate the skill of a machine learning model on unseen data. That is, to use a limited sample in order to estimate how the model is expected to perform in general when used to make predictions on data not used during the training of the model.
A test set should still be held out for final evaluation, but the validation set is no longer needed when doing CV. In the basic approach, called *k*-fold CV, the training set is split into *k* smaller sets (other approaches are described below, but generally follow the same principles). The following procedure is followed for each of the *k* "folds":

- A model is trained using k−1 of the folds as training data;
- the resulting model is validated on the remaining part of the data (i.e., it is used as a test set to compute a performance measure such as accuracy).

The performance measure reported by *k*-fold cross-validation is then the average of the values computed in the loop. This approach can be computationally expensive, but does not waste too much data (as is the case when fixing an arbitrary validation set), which is a major advantage in problems such as inverse inference where the number of samples is very small.

10. What is hyper parameter tuning in machine learning and why it is done?

Answer: In machine learning, hyperparameter optimization or tuning is the problem of choosing a set of optimal hyperparameters for a learning algorithm. A hyperparameter is a parameter whose value is used to control the learning process. By contrast, the values of other parameters (typically node weights) are learned.
Hyperparameters are important because they directly control the behaviour of the training algorithm and have a significant impact on the performance of the model is being trained. "A good choice of hyperparameters can really make an algorithm shine". ... Easy to manage a large set of experiments for hyperparameter tuning.

11. What issues can occur if we have a large learning rate in Gradient Descent?

Answer: When the learning rate is too large, gradient descent can inadvertently increase rather than decrease the training error. When the learning rate is too small, training is not only slower, but may become permanently stuck with a high training error.

12. What is bias-variance trade off in machine learning?

Answer: Bias is the simplifying assumptions made by the model to make the target function easier to approximate. **Variance** is the amount that the estimate of the target function will change given different training data. **Trade-off** is tension between the error introduced by the bias and the variance.
The bias–variance trade-off is the property of a set of predictive models whereby models with a lower bias in parameter estimation have a higher variance of the parameter estimates across samples, and vice versa.

13. What is the need of regularization in machine learning?

Answer: Regularisation is a technique used to reduce the errors by fitting the function appropriately on the given training set and avoid overfitting.
The commonly used regularisation techniques are:

1. L1 regularisation (LASSO (Least Absolute Shrinkage and Selection Operator))
2. L2 regularisation (Ridge regression)
3. Dropout regularisation

14. Differentiate between Adaboost and Gradient Boosting?

Answer:

AdaBoost	GradientBoost
Both AdaBoost and Gradient Boost use a base weak learner and they try to boost the performance of a weak learner by iteratively shifting the focus towards problematic observations that were difficult to predict. At the end, a strong learner is formed by addition (or weighted addition) of the weak learners.	
In AdaBoost, shift is done by up-weighting observations that were misclassified before.	Gradient boost identifies difficult observations by large residuals computed in the previous iterations.
In AdaBoost "shortcomings" are identified by high-weight data points.	In Gradientboost "shortcomings" are identified by gradients.
Exponential loss of AdaBoost gives more weights for those samples fitted worse.	Gradient boost further dissect error components to bring in more explanation.
AdaBoost is considered as a special case of Gradient boost in terms of loss function, in which exponential losses.	Concepts of gradients are more general in nature.

15. Can we use Logistic Regression for classification of Non-Linear Data? If not, why?

Answer: Logistic regression is known and used as a linear classifier. It is used to come up with a hyperplane in feature space to separate observations that belong to a class from all the other observations that do not belong to that class. The decision boundary is thus linear. Robust and efficient implementations are readily available (e.g. scikit-learn) to use logistic regression as a linear classifier.