Support Vector Machine Interview Questions

We believe that you have learned both theoritical and practical knowledge on Naive Bayes classification algorithm through your assignment.

So let's test your knowledge here. This will help you to be prepared for interviews too!

Best with Quest

1. What are Support Vector Machines (SVMs)? What is the basic principle of a Support Vector Machine?

☞ SVM is a supervised machine learning algorithm that works on both classification and regression problem statements.

☞ For classification problem statements, it tries to differentiate data points of different classes by finding a hyperplane that maximizes the margin between the classes in the training data.

☞ In simple words, SVM tries to choose the hyperplane which separates the data points as widely as possible since this margin maximization improves the model's accuracy on the test or the unseen data.

It's aimed at finding an optimal hyperplane that is linearly separable, and for the dataset which is not directly linearly separable, it extends its formulation by transforming the original data to map into a new space, which is also called kernel trick.

2. What are Support Vectors in SVMs? What are hard margin and soft Margin SVMs?

☞ Support vectors are those instances that are located on the margin itself. For SVMS, the decision boundary is entirely determined by using only the support vectors.
☞ Any instance that is not a support vector (not on the margin boundaries) has no influence whatsoever; you could remove them or add more instances, or move them around, and as long as they stay off the margin they won't affect the decision boundary.
☞ For computing the predictions, only the support vectors are involved, not the whole training set.

☞ Hard margin SVMs work only if the data is linearly separable and these types of SVMs are quite sensitive to the outliers.
☞ But our main objective is to find a good balance between keeping the margins as large as possible and limiting the margin violation i.e. instances that end up in the middle of margin or even on the wrong side, and this method is called soft margin SVM

3. What is the "Kernel trick"?

```
A function is called kernel if there exist a function φ that maps a and b into another
space such that K(a, b) = φ(a)T · φ(b). So you can use K as a kernel since you just know
that a mapping φ exists, even if you don’t know what φ function is. These are the very
good things about kernels.

Some of the kernel functions are as follows:

👉 Polynomial Kernel: These are the kernel functions that represent the similarity of
vectors in a feature space over polynomials of original variables.

👉 Gaussian Radial Basis Function (RBF) kernel:  Gaussian RBF kernel maps each training
instance to an infinite-dimensional space, therefore it’s a good thing that you don’t need
to perform the mapping.
```

4. What is the role of the C hyper-parameter in SVM? Does it affect the bias/variance trade-off?

```
👉 The balance between keeping the margins as large as possible and limiting the margin
violation is controlled by the C parameter: a small value leads to a wider street but more
margin violation and a higher value of C makes fewer margin violations but ends up with a
smaller margin and overfitting.

👉 Here thing becomes a little complex as we have conflicting objectives of making the
slack variables as small as possible to reduce margin violation and make W as small as
possible to increase the margin. This is where the role of the C hyperparameter comes in
which allows us to define the trade-off between these two objectives
```

5. What is a slack variable?

If you train an SVM classifier with an RBF kernel. It seems to underfit the training dataset: should you increase or decrease the hyper-parameter γ (gamma)? What about the C hyper-parameter?

```
👉 To meet the soft margin objective, we need to introduce a slack variable ε>=0 for each
sample; it measures how much any particular instance is allowed to violate the margin.

👉 Here thing becomes little complex as we have conflicting objectives of making the
slack variables as small as possible to reduce margin violation and make w (weight matrix)
as small as possible to increase the margin. This is where the role of the C
hyperparameter comes which allows us to define the trade-off between these two objectives.


If we trained an SVM classifier using a Radial Basis Function (RBF) kernel, then it
underfits the training set, so there might be too much regularization. To decrease it, you
need to increase the gamma or C hyper-parameter.
```

6. What is Polynomial kernel? Explain about SVM Regression?

```
Polynomial kernel is a kernel function commonly used with support vector machines (SVMs)
and other kernelized models, that represents the similarity of vectors (training samples)
in a feature space over polynomials of the original variables, allowing learning of non-
linear models.
```

```
The Support Vector Regression (SVR) uses the same principles as the SVM for
classification, with only a few minor differences. First of all, because the output is a
real number it becomes very difficult to predict the information at hand, which has
infinite possibilities. In the case of regression, a margin of tolerance (epsilon) is set
in approximation to the SVM
```

7. Give some situations where you will use an SVM over a RandomForest Machine Learning algorithm. SVM being a large margin classifier, is it influenced by outliers?

```
The main reason to use an SVM instead is that the problem might not be linearly separable.
In that case, we will have to use an SVM with a non-linear kernel (e.g. RBF).
Another related reason to use SVMs is if you are in a higher-dimensional space. For
example, SVMs have been reported to work better for text classification.

Yes, if C is large, otherwise not.
```

8. Can we apply the kernel trick to logistic regression? Why is it not used in practice then? What is the difference between logistic regression and SVM without a kernel? Does SVM give any probabilistic output?

```
Logistic Regression is computationally more expensive than SVM — O(N³) vs O(N²k) where k
is the number of support vectors.
The classifier in SVM is designed such that it is defined only in terms of the support
vectors, whereas in Logistic Regression, the classifier is defined over all the points and
not just the support vectors. This allows SVMs to enjoy some natural speed-ups (in terms
of efficient code-writing) that is hard to achieve for Logistic Regression.

They differ only in the implementation . SVM is much more efficient and has good
optimization packages.


SVMs do not directly provide probability estimates, these are calculated using an
expensive five-fold cross-validation
```

9. What is regularization? How is an SVM algorithm regularized? Is it L1 or L2 regularization in nature?

Why the SVM algorithm is known to be memory efficient?

```
‘C parameter’ is actually the regularization parameter. The C parameter is multiplied by
the sum of errors thus the regularization, in general, is L1 in nature. But if the cost
function is modified in such a way that the cost of SVM is C multiplied by the sum of
squares of the error the regularization becomes L2 in nature.

Not all the data points are required to make the decision, once the support vectors are
decided only the support vectors and equation of the hyperplane is required to make the
decision.
```

10. Suppose you are using RBF kernel in SVM with high Gamma value. What does this signify? What is the difference between logistic regression and SVM

The gamma parameter in SVM tuning signifies the influence of points either near or far away from the hyperplane.
For a low gamma, the model will be too constrained and include all points of the training dataset, without really capturing the shape.
For a higher gamma, the model will capture the shape of the dataset well.

Logistic regression assumes that the predictors aren’t sufficient to determine the response variable, but determine a probability that is a logistic function of a linear combination of them. If there’s a lot of noise, logistic regression (usually fit with maximum-likelihood techniques) is a great technique.
On the other hand, there are problems where you have thousands of dimensions and the predictors do nearly-certainly determine the response, but in some hard-to-explicitly-program way. An example would be image recognition. If you have a grayscale image, 100 by 100 pixels, you have 10,000 dimensions already. With various basis transforms (kernel trick) you will be able to get a linear separator of the data.
Non-regularized logistic regression techniques don’t work well (in fact, the fitted coefficients diverge) when there’s a separating hyperplane, because the maximum likelihood is achieved by any separating plane, and there’s no guarantee that you’ll get the best one. What you get is an extremely confident model with poor predictive power near the margin.
SVMs get you the best separating hyperplane, and they’re efficient in high dimensional spaces. They’re similar to regularization in terms of trying to find the lowest-normed vector that separates the data, but with a margin condition that favors choosing a good hyperplane. A hard-margin SVM will find a hyperplane that separates all the data (if one exists) and fail if there is none; soft-margin SVMs (generally preferred) do better when there’s noise in the data.
Additionally, SVMs only consider points near the margin (support vectors). Logistic regression considers all the points in the data set. Which you prefer depends on your problem.
Logistic regression is great in a low number of dimensions and when the predictors don’t suffice to give more than a probabilistic estimate of the response. SVMs do better when there’s a higher number of dimensions, and especially on problems where the predictors do certainly (or near-certainly) determine the responses.

Now Rest with this Quest :)