SVM Interview Questions

Answer 1

Relationship between polynomial functions and kernel functions in machine learning algorithms

- Polynomial functions and kernel functions are both used in machine learning algorithms, particularly in the context of support vector machines (SVMs).

Kernal Function:

A kernel function is used to map the original input data to a higher-dimensional space, where it is more likely that a hyperplane can be found to separate the classes.

Polynomial functions:

The polynomial kernel function essentially allows the SVM to use a polynomial function to separate the data in the high-dimensional space, without explicitly computing the coordinates of the data points in that space.

Answer 2

Implementing an SVM with a polynomial kernel is straightforward using the SVC (Support Vector Classification) class.

In [ ]:
```
from sklearn import datasets
from sklearn.model_selection import train_test_split
from sklearn.svm import SVC
from sklearn.metrics import accuracy_score

# Load the iris dataset
iris = datasets.load_iris()

# Split the data into training and test sets
X_train, X_test, y_train, y_test = train_test_split(iris.data, iris.target, test

# Create an instance of the SVM with a polynomial kernel
svm_poly = SVC(kernel='poly', degree=3)

# Train the SVM on the training data
svm_poly.fit(X_train, y_train)

# Use the trained SVM to predict the classes of the test data
```

```
y_pred = svm_poly.predict(X_test)

# Compute the accuracy of the SVM on the test data
accuracy = accuracy_score(y_test, y_pred)
print("Accuracy: {:.2f}%".format(accuracy * 100))
```

```
Accuracy: 91.67%
```

Answer 3

Increasing the value of epsilon affect the number of support vectors in SVR

Epsilon (ε):

The parameter epsilon (ε) determines the width of the margin of tolerance around the predicted function. Specifically, epsilon controls the size of the "epsilon-tube" around the regression line, within which errors are not penalized.

Affect:

- The number of support vectors in SVR can be affected by the value of epsilon.
- Increasing the value of epsilon can lead to an increase or decrease in the number of support vectors, depending on the specific dataset and the value of other parameters such as the kernel function and regularization parameter.
- It is important to note that while increasing epsilon can result in a larger number of support vectors, this can also lead to overfitting,

---

Answer 5

Effect of Kernel Function, C parameter, epsilon parameter, and gamma parameter on Support vector Regression

1 Kernel Function:

- Kernal Function determined the how input data is transformed into high dimensional feature space in which linear regression is performed
- Some common kernel functions are linear , polynomial, RBF, and Sigmoid.
- For Example if the Dataset have non - linear relationship between betweeen features and target variables then we will prefer RBF and polynomial than linear kernel

2 C Parameter:

- The C parameter controls the trade-off between maximizing the margin (i.e., the distance between the regression line and the support vectors) and minimizing the error.
- A smaller value of C will allow for more errors but a wider margin, while a larger value of C will result in a narrower margin but fewer errors
- If the dataset has a large number of outliers, a smaller value of C may be more appropriate to allow for more errors, while a larger value of C may be better for a dataset with fewer outliers.

3 Epsilon Parameter:

- The epsilon parameter (ε) controls the width of the margin of tolerance around the predicted function, within which errors are not penalized.
- A larger value of epsilon allows for more errors within the margin, while a smaller value of epsilon results in fewer errors. If the dataset has a lot of noise or variability, a larger value of epsilon may be more appropriate to allow for more errors, while a smaller value of epsilon may be better for a dataset with less noise.

4 Gamma Parameter:

- The gamma parameter determines the shape of the decision boundary and the influence of each training example.
- A smaller value of gamma results in a more gradual change in the decision boundary, while a larger value of gamma results in a more abrupt change.
- If the dataset has many features or is high-dimensional, a smaller value of gamma may be more appropriate to prevent overfitting, while a larger value of gamma may be better for a dataset with fewer features or is low-dimensional.

Answer 5

ASSIGNMENT

In [ ]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import plotly.express as  exp
```

In [ ]:
```
# laod the dataset
from sklearn.datasets import load_breast_cancer
from sklearn.model_selection import train_test_split
from sklearn.svm import SVC
from sklearn.preprocessing import StandardScaler
from sklearn.metrics import accuracy_score
from sklearn.model_selection import GridSearchCV
import joblib
```

```
data = load_breast_cancer()
```

In [ ]:
```
# Split dataset into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(data.data, data.target, test
```

In [ ]:
```
# Preprocess data by scaling it
scaler = StandardScaler()
X_train = scaler.fit_transform(X_train)
X_test = scaler.transform(X_test)
```

In [ ]:
```
# Create instance of SVC classifier and train it on training data
clf = SVC(kernel='rbf', random_state=42)
clf.fit(X_train, y_train)
clf
```

Out[ ]:
```
SVC
SVC(random_state=42)
```

In [ ]:
```
 # Use trained classifier to predict labels of testing data
y_pred = clf.predict(X_test)
y_pred
```

Out[ ]:
```
array([1, 0, 0, 1, 1, 0, 0, 0, 0, 1, 1, 0, 1, 0, 1, 0, 1, 1, 1, 0, 1, 1,
       0, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 0, 1, 0, 1, 1, 0, 1, 1,
       1, 1, 1, 1, 1, 1, 0, 0, 1, 1, 1, 1, 1, 0, 0, 1, 1, 0, 0, 1, 1, 1,
       0, 0, 1, 1, 0, 0, 1, 0, 1, 1, 1, 1, 1, 1, 0, 1, 0, 0, 0, 0, 0, 0,
       1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 1, 0, 0, 1, 0, 0, 1, 1, 1, 0, 1, 1,
       0, 1, 1, 0, 0, 0, 1, 1, 1, 0, 1, 1, 1, 0, 1, 0, 0, 1, 1, 0, 0, 0,
       1, 1, 1, 0, 1, 1, 1, 0, 1, 0, 1, 1, 0, 1, 0, 0, 0, 1, 0, 1, 1, 1,
       1, 0, 0, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 0, 1])
```

In [ ]:
```
# Evaluate performance of classifier using accuracy metric
accuracy = accuracy_score(y_test, y_pred)
print(f"Accuracy: {accuracy}")
```

```
Accuracy: 0.9766081871345029
```

In [ ]:
```
# Tune hyperparameters of SVC classifier using GridSearchCV
param_grid = {
    'C': [0.1, 1, 10, 100],
    'gamma': ['scale', 'auto'],
    'kernel': ['rbf', 'poly', 'sigmoid']
}
```

In [ ]:
```
grid_search = GridSearchCV(clf, param_grid=param_grid, cv=5)
grid_search.fit(X_train, y_train)
print(f"Best parameters: {grid_search.best_params_}")
print(f"Best score: {grid_search.best_score_}")
```

```
Best parameters: {'C': 100, 'gamma': 'scale', 'kernel': 'rbf'}
Best score: 0.9673417721518988
```

In [ ]:
```
# Train tuned classifier on entire dataset
clf_tuned = SVC(**grid_search.best_params_)
clf_tuned.fit(data.data, data.target)
```

```
# Save trained classifier to a file
joblib.dump(clf_tuned, 'svm_classifier.pkl')
```

Out[ ]:
```
['svm_classifier.pkl']
```

In [ ]:
```

```