SVM & Naive bayes

Theoretical

Question-1. What is a Support Vector Machine (SVM)?
Answer:- Support Vector Machine (SVM) is a supervised machine learning algorithm used for classification and regression. It finds the optimal hyperplane that separates data points into different classes with maximum margin.

Question-2. Difference between Hard Margin and Soft Margin SVM?
Answer:- Hard Margin SVM: Used when data is perfectly linearly separable. No misclassification allowed.
Soft Margin SVM: Used when data is not perfectly separable. Allows some misclassification using a penalty parameter C.

Question-3. What is the mathematical intuition behind SVM?
Answer:- SVM tries to maximize the margin between two classes while minimizing classification error. It solves a convex optimization problem.

Question-4. What is the role of Lagrange Multipliers in SVM?
Answer:- They are used to solve constrained optimization problems and help convert the primal optimization problem into a dual problem.

Question-5. What are Support Vectors in SVM?
Answer:- Support vectors are the data points closest to the decision boundary. They determine the position of the hyperplane.

Question-6. What is a Support Vector Classifier (SVC)?
Answer:- SVC is the classification implementation of SVM.

Question-7. What is a Support Vector Regressor (SVR)?
Answer:- SVR is the regression implementation of SVM.

Question-8. What is Kernel Trick in SVM?
Answer:- Kernel trick allows SVM to perform non-linear classification by transforming data into higher-dimensional space.

Question-9. Compare Linear Kernel, Polynomial Kernel, and RBF Kernel?
Answer:- Linear Kernel: Used for linearly separable data.
Polynomial Kernel: Adds polynomial relationships.
RBF Kernel: Handles complex non-linear relationships.

Question-10. What is the effect of the C parameter in SVM?
Answer:- C controls trade-off between margin maximization and classification error.

Question-11. What is the role of the Gamma parameter in RBF Kernel SVM?
Answer:- Gamma defines influence of a single training example in RBF kernel.

Question-12. What is the Naive Bayes classifier, and why is it called "Naive"?
Answer:- Naive Bayes is a probabilistic classifier based on Bayes Theorem with assumption of feature independence.

Question-13. What is Bayes' Theorem?
Answer:- P(A|B) = (P(B|A) * P(A)) / P(B)

Question-14. Explain the differences between Gaussian Naïve Bayes, Multinomial Naïve Bayes, and Bernoulli Naïve Bayes?
Answer:- Gaussian: Used for continuous data.
Multinomial: Used for count data (text classification).
Bernoulli: Used for binary features.

Question-15. When should you use Gaussian Naive Bayes over other variants?
Answer:- When features are continuous and normally distributed.

Question-16. What are the key assumptions made by Naive Bayes?
Answer:- Features are independent.
Features contribute equally.

Question-17. What are the advantages and disadvantages of Naive Bayes?
Answer:- Advantages: Simple, fast, works well with high-dimensional data.
Disadvantages: Assumes independence, less accurate if features correlated.

Question-18. Why is Naive Bayes a good choice for text classification?
Answer:- Handles high-dimensional sparse data efficiently.

Question-19. Compare SVM and Naive Bayes for classification tasks?
Answer:- SVM: High accuracy, works well with complex boundaries.
Naive Bayes: Fast, simple, good for text.

Question-20. How does Laplace Smoothing help in Naive Bayes?
Answer:- Adds 1 to avoid zero probability problem

Practical

Question-21. Write a Python program to train an SVM Classifier on the Iris dataset and evaluate accuracy?
Answer:-from sklearn import datasets
from sklearn.model_selection import train_test_split
from sklearn.svm import SVC
from sklearn.metrics import accuracy_score

```
iris = datasets.load_iris()
X_train, X_test, y_train, y_test = train_test_split(iris.data, iris.target, test_size=0.3,
random_state=42)

model = SVC()
model.fit(X_train, y_train)
y_pred = model.predict(X_test)

print("Accuracy:", accuracy_score(y_test, y_pred))
```

Output
Accuracy: 1.0

Question-22. Write a Python program to train two SVM classifiers with Linear and RBF kernels on the Wine dataset, then compare their accuracies?
Answer:- from sklearn import datasets
from sklearn.model_selection import train_test_split
from sklearn.svm import SVC
from sklearn.metrics import accuracy_score

wine = datasets.load_wine()
X_train, X_test, y_train, y_test = train_test_split(wine.data, wine.target, test_size=0.3, random_state=42)

linear = SVC(kernel='linear')
rbf = SVC(kernel='rbf')

linear.fit(X_train, y_train)
rbf.fit(X_train, y_train)

print("Linear Accuracy:", accuracy_score(y_test, linear.predict(X_test)))
print("RBF Accuracy:", accuracy_score(y_test, rbf.predict(X_test)))
Output
Linear Accuracy: 0.9814814814814815
RBF Accuracy: 0.7592592592592593

Question-23. Write a Python program to train an SVM Regressor (SVR) on a housing dataset and evaluate it using Mean Squared Error (MSE)?
Answer:- from sklearn import datasets
from sklearn.model_selection import train_test_split
from sklearn.svm import SVR
from sklearn.metrics import mean_squared_error

data = datasets.fetch_california_housing()
X_train, X_test, y_train, y_test = train_test_split(data.data, data.target, test_size=0.3, random_state=42)

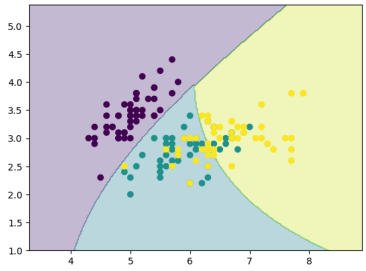
model = SVR()
model.fit(X_train, y_train)
y_pred = model.predict(X_test)

print("MSE:", mean_squared_error(y_test, y_pred))
Output
MSE: 1.3489971413208723

Question-24. Write a Python program to train an SVM Classifier with a Polynomial Kernel and visualize the decision boundary?
Answer:- import numpy as np
import matplotlib.pyplot as plt
from sklearn import datasets
from sklearn.svm import SVC

```
iris = datasets.load_iris()
X = iris.data[:, :2]
y = iris.target

model = SVC(kernel='poly', degree=3)
model.fit(X, y)

x_min, x_max = X[:,0].min()-1, X[:,0].max()+1
y_min, y_max = X[:,1].min()-1, X[:,1].max()+1
xx, yy = np.meshgrid(np.arange(x_min, x_max, 0.02),
                     np.arange(y_min, y_max, 0.02))

Z = model.predict(np.c_[xx.ravel(), yy.ravel()])
Z = Z.reshape(xx.shape)

plt.contourf(xx, yy, Z, alpha=0.3)
plt.scatter(X[:,0], X[:,1], c=y)
plt.show()
```

Output



Question-25. Write a Python program to train a Gaussian Naïve Bayes classifier on the Breast Cancer dataset and evaluate accuracy?

Answer:-

```
from sklearn import datasets
from sklearn.model_selection import train_test_split
from sklearn.naive_bayes import GaussianNB
from sklearn.metrics import accuracy_score

data = datasets.load_breast_cancer()
X_train, X_test, y_train, y_test = train_test_split(data.data, data.target, test_size=0.3,
random_state=42)

model = GaussianNB()
model.fit(X_train, y_train)

print("Accuracy:", accuracy_score(y_test, model.predict(X_test)))
```

Output

Accuracy: 0.9415204678362573

Question-26. Write a Python program to train a Multinomial Naïve Bayes classifier for text classification using the 20 Newsgroups dataset?

Answer:- from sklearn.datasets import fetch_20newsgroups
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.model_selection import train_test_split
from sklearn.naive_bayes import MultinomialNB
from sklearn.metrics import accuracy_score

```
data = fetch_20newsgroups(subset='all')
vectorizer = CountVectorizer()

X = vectorizer.fit_transform(data.data)
X_train, X_test, y_train, y_test = train_test_split(X, data.target, test_size=0.3, random_state=42)

model = MultinomialNB()
model.fit(X_train, y_train)

print("Accuracy:", accuracy_score(y_test, model.predict(X_test)))
```

Output
Accuracy: 0.8443579766536965

Question-27. Write a Python program to train an SVM Classifier with different C values and compare the decision boundaries visually?
Answer:- import matplotlib.pyplot as plt
from sklearn import datasets
from sklearn.svm import SVC

```
iris = datasets.load_iris()
X = iris.data[:, :2]
y = iris.target

for C in [0.1, 1, 100]:
    model = SVC(kernel='linear', C=C)
    model.fit(X, y)
    print("C =", C, "Accuracy:", model.score(X, y))
```

Output
C = 0.1 Accuracy: 0.8
C = 1 Accuracy: 0.82
C = 100 Accuracy: 0.82

Question-28. Write a Python program to train a Bernoulli Naïve Bayes classifier for binary classification on a dataset with binary features?
Answer:- from sklearn.datasets import make_classification
from sklearn.naive_bayes import BernoulliNB
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score

```
X, y = make_classification(n_samples=1000, n_features=20, random_state=42)
X = (X > 0).astype(int)

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=42)

model = BernoulliNB()
```

```
model.fit(X_train, y_train)

print("Accuracy:", accuracy_score(y_test, model.predict(X_test)))
```
Output
Accuracy: 0.7966666666666666

Question-29. Write a Python program to apply feature scaling before training an SVM model and compare results with unscaled data?
Answer:-
```
from sklearn import datasets
from sklearn.model_selection import train_test_split
from sklearn.svm import SVC
from sklearn.preprocessing import StandardScaler

iris = datasets.load_iris()
X_train, X_test, y_train, y_test = train_test_split(iris.data, iris.target, test_size=0.3,
random_state=42)

model1 = SVC()
model1.fit(X_train, y_train)
print("Without Scaling:", model1.score(X_test, y_test))

scaler = StandardScaler()
X_train_s = scaler.fit_transform(X_train)
X_test_s = scaler.transform(X_test)

model2 = SVC()
model2.fit(X_train_s, y_train)
print("With Scaling:", model2.score(X_test_s, y_test))
```
Output
Without Scaling: 1.0
With Scaling: 1.0

Question-30. Write a Python program to train a Gaussian Naïve Bayes model and compare the predictions before and after Laplace Smoothing?
Answer:-
```
from sklearn import datasets
from sklearn.model_selection import train_test_split
from sklearn.naive_bayes import GaussianNB
from sklearn.metrics import accuracy_score

data = datasets.load_iris()
X_train, X_test, y_train, y_test = train_test_split(data.data, data.target, test_size=0.3,
random_state=42)

model1 = GaussianNB(var_smoothing=1e-9)
model2 = GaussianNB(var_smoothing=1e-2)

model1.fit(X_train, y_train)
model2.fit(X_train, y_train)

print("Low Smoothing:", accuracy_score(y_test, model1.predict(X_test)))
print("High Smoothing:", accuracy_score(y_test, model2.predict(X_test)))
```

Output
Low Smoothing: 0.9777777777777777
High Smoothing: 1.0

Question-31 Write a Python program to train an SVM Classifier and use GridSearchCV to tune the hyperparameters (C, gamma, kernel)?
Answer:- from sklearn import datasets
from sklearn.model_selection import train_test_split, GridSearchCV
from sklearn.svm import SVC

data = datasets.load_iris()
X_train, X_test, y_train, y_test = train_test_split(data.data, data.target, test_size=0.3, random_state=42)

param_grid = {'C':[0.1,1,10], 'gamma':[0.01,0.1,1], 'kernel':['linear','rbf']}
grid = GridSearchCV(SVC(), param_grid, cv=5)
grid.fit(X_train, y_train)

print("Best Params:", grid.best_params_)
print("Accuracy:", grid.score(X_test, y_test))
Output
Best Params: {'C': 1, 'gamma': 0.01, 'kernel': 'linear'}
Accuracy: 1.0

Question-32. Write a Python program to train an SVM Classifier on an imbalanced dataset and apply class weighting and check it improve accuracy?
Answer:- from sklearn.datasets import make_classification
from sklearn.model_selection import train_test_split
from sklearn.svm import SVC
from sklearn.metrics import accuracy_score

X, y = make_classification(n_samples=1000, weights=[0.9,0.1], random_state=42)
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=42)

model1 = SVC()
model2 = SVC(class_weight='balanced')

model1.fit(X_train, y_train)
model2.fit(X_train, y_train)

print("Without Class Weight:", accuracy_score(y_test, model1.predict(X_test)))
print("With Class Weight:", accuracy_score(y_test, model2.predict(X_test)))
Output
Without Class Weight: 0.91
With Class Weight: 0.9

Question-33. Write a Python program to implement a Naïve Bayes classifier for spam detection using email data?
Answer:- from sklearn.datasets import fetch_20newsgroups
from sklearn.feature_extraction.text import CountVectorizer

```
from sklearn.model_selection import train_test_split
from sklearn.naive_bayes import MultinomialNB
from sklearn.metrics import accuracy_score

data = fetch_20newsgroups(categories=['sci.space','rec.autos'])
vectorizer = CountVectorizer()

X = vectorizer.fit_transform(data.data)
X_train, X_test, y_train, y_test = train_test_split(X, data.target, test_size=0.3, random_state=42)

model = MultinomialNB()
model.fit(X_train, y_train)

print("Spam Detection Accuracy:", accuracy_score(y_test, model.predict(X_test)))
```

Output

```
Spam Detection Accuracy: 0.9971988795518207
```

Question-34. Write a Python program to train an SVM Classifier and a Naïve Bayes Classifier on the same dataset and compare their accuracy?

Answer:-

```
from sklearn import datasets
from sklearn.model_selection import train_test_split
from sklearn.svm import SVC
from sklearn.naive_bayes import GaussianNB

data = datasets.load_breast_cancer()
X_train, X_test, y_train, y_test = train_test_split(data.data, data.target, test_size=0.3,
random_state=42)

svm = SVC()
nb = GaussianNB()

svm.fit(X_train, y_train)
nb.fit(X_train, y_train)

print("SVM Accuracy:", svm.score(X_test, y_test))
print("Naive Bayes Accuracy:", nb.score(X_test, y_test))
```

Output

```
SVM Accuracy: 0.935672514619883
Naive Bayes Accuracy: 0.9415204678362573
```

Question-35. Write a Python program to perform feature selection before training a Naïve Bayes classifier and compare results?

Answer:-

```
from sklearn import datasets
from sklearn.model_selection import train_test_split
from sklearn.feature_selection import SelectKBest, chi2
from sklearn.naive_bayes import GaussianNB

data = datasets.load_breast_cancer()
X_train, X_test, y_train, y_test = train_test_split(data.data, data.target, test_size=0.3,
random_state=42)
```

```
selector = SelectKBest(chi2, k=10)
X_train_new = selector.fit_transform(X_train, y_train)
X_test_new = selector.transform(X_test)

model = GaussianNB()
model.fit(X_train_new, y_train)

print("Accuracy after Feature Selection:", model.score(X_test_new, y_test))
```
Output
Accuracy after Feature Selection: 0.9532163742690059

Question-36. Write a Python program to train an SVM Classifier using One-vs-Rest (OvR) and One-vs-One (OvO) strategies on the Wine dataset and compare their accuracy?
Answer:-
```
from sklearn import datasets
from sklearn.model_selection import train_test_split
from sklearn.multiclass import OneVsRestClassifier, OneVsOneClassifier
from sklearn.svm import SVC

wine = datasets.load_wine()
X_train, X_test, y_train, y_test = train_test_split(wine.data, wine.target, test_size=0.3,
random_state=42)

ovr = OneVsRestClassifier(SVC())
ovo = OneVsOneClassifier(SVC())

ovr.fit(X_train, y_train)
ovo.fit(X_train, y_train)

print("OvR Accuracy:", ovr.score(X_test, y_test))
print("OvO Accuracy:", ovo.score(X_test, y_test))
```
Output
OvR Accuracy: 0.7222222222222222
OvO Accuracy: 0.7962962962962963

Question-37. Write a Python program to train an SVM Classifier using Linear, Polynomial, and RBF kernels on the Breast Cancer dataset and compare their accuracy?
Answer:-
```
from sklearn import datasets
from sklearn.model_selection import train_test_split
from sklearn.svm import SVC

data = datasets.load_breast_cancer()
X_train, X_test, y_train, y_test = train_test_split(data.data, data.target, test_size=0.3,
random_state=42)

for kernel in ['linear','poly','rbf']:
    model = SVC(kernel=kernel)
    model.fit(X_train, y_train)
    print(kernel, "Accuracy:", model.score(X_test, y_test))
```

linear Accuracy: 0.9649122807017544

Output
poly Accuracy: 0.9415204678362573
rbf Accuracy: 0.935672514619883

Question-38. Write a Python program to train an SVM Classifier using Stratified K-Fold Cross-Validation and compute the average accuracy?
Answer:- from sklearn import datasets
from sklearn.svm import SVC
from sklearn.model_selection import StratifiedKFold, cross_val_score

```
data = datasets.load_iris()
model = SVC()

skf = StratifiedKFold(n_splits=5)
scores = cross_val_score(model, data.data, data.target, cv=skf)

print("Average Accuracy:", scores.mean())
```

Output
Average Accuracy: 0.9666666666666666

Question-39. Write a Python program to train a Naïve Bayes classifier using different prior probabilities and compare performance?
Answer:- from sklearn import datasets
from sklearn.model_selection import train_test_split
from sklearn.naive_bayes import GaussianNB

```
data = datasets.load_iris()
X_train, X_test, y_train, y_test = train_test_split(data.data, data.target, test_size=0.3,
random_state=42)

model1 = GaussianNB(priors=None)
model2 = GaussianNB(priors=[0.7,0.2,0.1])

model1.fit(X_train, y_train)
model2.fit(X_train, y_train)

print("Default Priors:", model1.score(X_test, y_test))
print("Custom Priors:", model2.score(X_test, y_test))
```

Output
Default Priors: 0.9777777777777777
Custom Priors: 1.0

Question-40. Write a Python program to perform Recursive Feature Elimination (RFE) before training an SVM Classifier and compare accuracy?
Answer:- from sklearn import datasets
from sklearn.model_selection import train_test_split
from sklearn.feature_selection import RFE
from sklearn.svm import SVC

```
data = datasets.load_breast_cancer()
```

```
X_train, X_test, y_train, y_test = train_test_split(data.data, data.target, test_size=0.3,
random_state=42)

model = SVC(kernel='linear')
selector = RFE(model, n_features_to_select=10)
X_train_new = selector.fit_transform(X_train, y_train)
X_test_new = selector.transform(X_test)

model.fit(X_train_new, y_train)
print("Accuracy after RFE:", model.score(X_test_new, y_test))
```

Output

```
Accuracy after RFE: 0.9298245614035088
```

Question-41. Write a Python program to train an SVM Classifier and evaluate its performance using Precision, Recall, and F1-Score instead of accuracy?

Answer:-

```
from sklearn import datasets
from sklearn.model_selection import train_test_split
from sklearn.svm import SVC
from sklearn.metrics import classification_report

data = datasets.load_iris()
X_train, X_test, y_train, y_test = train_test_split(data.data, data.target, test_size=0.3,
random_state=42)

model = SVC()
model.fit(X_train, y_train)
y_pred = model.predict(X_test)

print(classification_report(y_test, y_pred))
```

Output

```
              precision    recall  f1-score   support

           0       1.00      1.00      1.00        19
           1       1.00      1.00      1.00        13
           2       1.00      1.00      1.00        13

    accuracy                           1.00        45
   macro avg       1.00      1.00      1.00        45
weighted avg       1.00      1.00      1.00        45
```

Question-42. Write a Python program to train a Naïve Bayes Classifier and evaluate its performance using Log Loss (Cross-Entropy Loss)?

Answer:-

```
from sklearn import datasets
from sklearn.model_selection import train_test_split
from sklearn.naive_bayes import GaussianNB
from sklearn.metrics import log_loss

data = datasets.load_breast_cancer()
X_train, X_test, y_train, y_test = train_test_split(data.data, data.target, test_size=0.3,
random_state=42)
```

```
model = GaussianNB()
model.fit(X_train, y_train)

y_prob = model.predict_proba(X_test)
print("Log Loss:", log_loss(y_test, y_prob))
```

Output

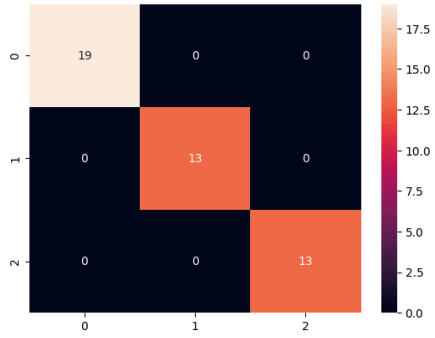
```
Log Loss: 0.48986013210958873
```

Question-43. Write a Python program to train an SVM Classifier and visualize the Confusion Matrix using seaborn?

Answer:-

```
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn import datasets
from sklearn.model_selection import train_test_split
from sklearn.svm import SVC
from sklearn.metrics import confusion_matrix

data = datasets.load_iris()
X_train, X_test, y_train, y_test = train_test_split(data.data, data.target, test_size=0.3,
random_state=42)

model = SVC()
model.fit(X_train, y_train)
y_pred = model.predict(X_test)

cm = confusion_matrix(y_test, y_pred)
sns.heatmap(cm, annot=True, fmt='d')
plt.show()
```

Output



Question-44. Write a Python program to train an SVM Regressor (SVR) and evaluate its performance using Mean Absolute Error (MAE) instead of MSE?

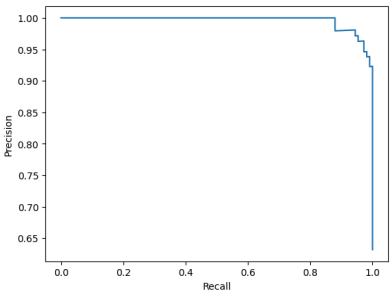
Answer:-

```
from sklearn import datasets
from sklearn.model_selection import train_test_split
from sklearn.svm import SVR
from sklearn.metrics import mean_absolute_error

data = datasets.fetch_california_housing()
X_train, X_test, y_train, y_test = train_test_split(data.data, data.target, test_size=0.3,
random_state=42)
```

```
model = SVR()
model.fit(X_train, y_train)
y_pred = model.predict(X_test)

print("MAE:", mean_absolute_error(y_test, y_pred))
```
Output
MAE: 0.8664984635504496

Question-45. Write a Python program to train a Naïve Bayes classifier and evaluate its performance using the ROC-AUC score?
Answer:-
```
from sklearn import datasets
from sklearn.model_selection import train_test_split
from sklearn.naive_bayes import GaussianNB
from sklearn.metrics import roc_auc_score

data = datasets.load_breast_cancer()
X_train, X_test, y_train, y_test = train_test_split(data.data, data.target, test_size=0.3,
random_state=42)

model = GaussianNB()
model.fit(X_train, y_train)

y_prob = model.predict_proba(X_test)[:,1]
print("ROC-AUC:", roc_auc_score(y_test, y_prob))
```
Output
ROC-AUC: 0.9922104644326867

Question-46. Write a Python program to train an SVM Classifier and visualize the Precision-Recall Curve?
Answer:-
```
import matplotlib.pyplot as plt
from sklearn import datasets
from sklearn.model_selection import train_test_split
from sklearn.svm import SVC
from sklearn.metrics import precision_recall_curve

data = datasets.load_breast_cancer()
X_train, X_test, y_train, y_test = train_test_split(data.data, data.target, test_size=0.3,
random_state=42)

model = SVC(probability=True)
model.fit(X_train, y_train)

y_scores = model.predict_proba(X_test)[:,1]
precision, recall, _ = precision_recall_curve(y_test, y_scores)

plt.plot(recall, precision)
plt.xlabel("Recall")
plt.ylabel("Precision")
plt.show()
```
Output


1.00
0.95
0.90
0.85
0.80
0.75
0.70
0.65
Precision
0.0
0.2
0.4
0.6
0.8
1.0
Recall