ASSIGNMENT 4

In [17]:
```python
import pandas as pd

# Try reading the CSV file with different encodings as it doesnt work with utf-8
encodings_to_try = ['utf-8', 'ISO-8859-1', 'cp1252']

for encoding in encodings_to_try:
    try:
        df = pd.read_csv(r"spam.csv", encoding=encoding)
        print("File Read Successfully with Encoding:", encoding)
        print(df)
        break
    except UnicodeDecodeError:
        print(f"Failed to read with encoding {encoding}. Trying the next one.")
```

```
Failed to read with encoding utf-8. Trying the next one.
File Read Successfully with Encoding: ISO-8859-1
        v1                                                 v2 Unnamed: 2  \
0      ham  Go until jurong point, crazy.. Available only ...        NaN
1      ham                      Ok lar... Joking wif u oni...        NaN
2     spam  Free entry in 2 a wkly comp to win FA Cup fina...        NaN
3      ham  U dun say so early hor... U c already then say...        NaN
4      ham  Nah I don't think he goes to usf, he lives aro...        NaN
...    ...                                                ...        ...
5567  spam  This is the 2nd time we have tried 2 contact u...        NaN
5568   ham              Will Ì_ b going to esplanade fr home?        NaN
5569   ham  Pity, * was in mood for that. So...any other s...        NaN
5570   ham  The guy did some bitching but I acted like i'd...        NaN
5571   ham                         Rofl. Its true to its name        NaN

     Unnamed: 3 Unnamed: 4
0           NaN        NaN
1           NaN        NaN
2           NaN        NaN
3           NaN        NaN
4           NaN        NaN
...         ...        ...
5567        NaN        NaN
5568        NaN        NaN
5569        NaN        NaN
5570        NaN        NaN
5571        NaN        NaN

[5572 rows x 5 columns]
```

Pre-processing

In [18]:
```python
df=df.drop(columns=["Unnamed: 2","Unnamed: 3","Unnamed: 4"])
print(df)
```

```
        v1                                                 v2
0      ham  Go until jurong point, crazy.. Available only ...
1      ham                      Ok lar... Joking wif u oni...
2     spam  Free entry in 2 a wkly comp to win FA Cup fina...
3      ham  U dun say so early hor... U c already then say...
4      ham  Nah I don't think he goes to usf, he lives aro...
...    ...                                                ...
5567  spam  This is the 2nd time we have tried 2 contact u...
5568   ham              Will Ì_ b going to esplanade fr home?
5569   ham  Pity, * was in mood for that. So...any other s...
5570   ham  The guy did some bitching but I acted like i'd...
5571   ham                         Rofl. Its true to its name

[5572 rows x 2 columns]
```

In [19]:
```python
from sklearn.preprocessing import LabelEncoder

df.dropna()

label_encoder=LabelEncoder()
df["v1"]=label_encoder.fit_transform(df["v1"])
```

Splitting data

In [29]:
```
from sklearn.model_selection import train_test_split
from sklearn.feature_extraction.text import CountVectorizer

y=df["v1"]
X=df["v2"]

X_train,X_test,y_train,y_test=train_test_split(X,y,test_size=0.25,random_state=42)

vectorizer=CountVectorizer()
X_train_vectorized=vectorizer.fit_transform(X_train)
X_test_vectorized=vectorizer.transform(X_test)
```

Classification using Naive Bayes, Decision Tree, SVM

In [30]:
```
from sklearn.naive_bayes import MultinomialNB
from sklearn.tree import DecisionTreeClassifier
from sklearn.svm import SVC

#Naive Bayes
nb_classifier=MultinomialNB()
nb_classifier.fit(X_train_vectorized,y_train)
y_pred=nb_classifier.predict(X_test_vectorized)


#Decision Tree Classifier
model=DecisionTreeClassifier(criterion="gini",max_depth=3,random_state=42)
model.fit(X_train_vectorized,y_train)
y_pred=model.predict(X_test_vectorized)


#SVM
svc_classifier=SVC(kernel="linear")
svc_classifier.fit(X_train_vectorized,y_train)
y_pred=svc_classifier.predict(X_test_vectorized)
```

Cross-validation for above models

In [33]:
```
from sklearn.model_selection import cross_val_score, GridSearchCV

nb_cv_scores = cross_val_score(nb_classifier, X_train_vectorized, y_train, cv=5, scoring='accuracy')
print(f"Naive Bayes Cross-Validation Scores: {nb_cv_scores}")
print(f"Mean Accuracy: {nb_cv_scores.mean()}\n")

dtc_cv_scores = cross_val_score(model, X_train_vectorized, y_train, cv=5, scoring='accuracy')
print(f"Decision tree Cross-Validation Scores: {dtc_cv_scores}")
print(f"Mean Accuracy: {dtc_cv_scores.mean()}\n")

svm_cv_scores = cross_val_score(svc_classifier, X_train_vectorized, y_train, cv=5, scoring='accuracy')
print(f"SVM Cross-Validation Scores: {svm_cv_scores}")
print(f"Mean Accuracy: {svm_cv_scores.mean()}\n")
```

```
Naive Bayes Cross-Validation Scores: [0.98444976 0.98803828 0.98684211 0.98205742 0.96886228]
Mean Accuracy: 0.9820499670515428

Decision tree Cross-Validation Scores: [0.95095694 0.93660287 0.93301435 0.93421053 0.94371257]
Mean Accuracy: 0.9396994527691028

SVM Cross-Validation Scores: [0.99162679 0.98803828 0.98205742 0.98325359 0.98443114]
Mean Accuracy: 0.985881442855915
```

Hypertuning for above models

In [37]:
```
nb_param_grid = {'alpha': [0.1, 0.5, 1.0, 2.0]}
nb_grid_search = GridSearchCV(nb_classifier, nb_param_grid, cv=5, scoring='accuracy')
nb_grid_search.fit(X_train_vectorized, y_train)
print(f"Best Naive Bayes Hyperparameters: {nb_grid_search.best_params_}")

dt_param_grid = {
    'criterion': ['gini', 'entropy'],
    'max_depth': [None, 5, 10, 15],
    'min_samples_split': [2, 5, 10],
    'min_samples_leaf': [1, 2, 4]
}

dt_grid_search = GridSearchCV(model, dt_param_grid, cv=5, scoring='accuracy')
```

```
dt_grid_search.fit(X_train_vectorized, y_train)
print(f"Best Decision Tree Hyperparameters: {dt_grid_search.best_params_}")

svm_param_grid = {'C': [0.1, 1, 10], 'kernel': ['linear', 'rbf']}
svm_grid_search = GridSearchCV(svc_classifier, svm_param_grid, cv=5, scoring='accuracy')
svm_grid_search.fit(X_train_vectorized, y_train)
print(f"Best SVM Hyperparameters: {svm_grid_search.best_params_}")
```

```
Best Naive Bayes Hyperparameters: {'alpha': 1.0}
Best Decision Tree Hyperparameters: {'criterion': 'gini', 'max_depth': None, 'min_samples_leaf': 1, 'min_samples_
split': 2}
Best SVM Hyperparameters: {'C': 1, 'kernel': 'linear'}
```