In [46]:
```
import pandas as pd
df=pd.read_csv("C:\\Users\\GPT BANTWAL\\Downloads\\Breast_Cancer_data (1).csv")
df
```

Out[46]:

	id	diagnosis	radius_mean	texture_mean	perimeter_mean	area_mean	smoothness_
0	842302	M	17.99	10.38	122.80	1001.0	0.
1	842517	M	20.57	17.77	132.90	1326.0	0.
2	84300903	M	19.69	21.25	130.00	1203.0	0.
3	84348301	M	11.42	20.38	77.58	386.1	0.
4	84358402	M	20.29	14.34	135.10	1297.0	0.
...	...	...	...	...	...	...	
564	926424	M	21.56	22.39	142.00	1479.0	0.
565	926682	M	20.13	28.25	131.20	1261.0	0.
566	926954	M	16.60	28.08	108.30	858.1	0.
567	927241	M	20.60	29.33	140.10	1265.0	0.
568	92751	B	7.76	24.54	47.92	181.0	0.

569 rows × 33 columns

In [ ]:
```
```

In [47]:
```
from sklearn.preprocessing import LabelEncoder
le=LabelEncoder()
df['diagnosis']=le.fit_transform(df['diagnosis'])
df['diagnosis']
```

Out[47]:
```
0      1
1      1
2      1
3      1
4      1
      ..
564    1
565    1
566    1
567    1
568    0
Name: diagnosis, Length: 569, dtype: int32
```

In [48]:
```
df
```

Out[48]:

	id	diagnosis	radius_mean	texture_mean	perimeter_mean	area_mean	smoothness_
0	842302	1	17.99	10.38	122.80	1001.0	0.
1	842517	1	20.57	17.77	132.90	1326.0	0.
2	84300903	1	19.69	21.25	130.00	1203.0	0.
3	84348301	1	11.42	20.38	77.58	386.1	0.
4	84358402	1	20.29	14.34	135.10	1297.0	0.
...	...	...	...	...	...	...	
564	926424	1	21.56	22.39	142.00	1479.0	0.
565	926682	1	20.13	28.25	131.20	1261.0	0.
566	926954	1	16.60	28.08	108.30	858.1	0.
567	927241	1	20.60	29.33	140.10	1265.0	0.
568	92751	0	7.76	24.54	47.92	181.0	0.

569 rows × 33 columns

In [49]:
```
x=df.drop(['diagnosis','Unnamed: 32'],axis=1)
x
```

Out[49]:

	id	radius_mean	texture_mean	perimeter_mean	area_mean	smoothness_mean	com
0	842302	17.99	10.38	122.80	1001.0	0.11840	
1	842517	20.57	17.77	132.90	1326.0	0.08474	
2	84300903	19.69	21.25	130.00	1203.0	0.10960	
3	84348301	11.42	20.38	77.58	386.1	0.14250	
4	84358402	20.29	14.34	135.10	1297.0	0.10030	
...	...	...	...	...	...	...	
564	926424	21.56	22.39	142.00	1479.0	0.11100	
565	926682	20.13	28.25	131.20	1261.0	0.09780	
566	926954	16.60	28.08	108.30	858.1	0.08455	
567	927241	20.60	29.33	140.10	1265.0	0.11780	
568	92751	7.76	24.54	47.92	181.0	0.05263	

569 rows × 31 columns

In [50]:
```
y=df['diagnosis']
y
```

Out[50]:
```
0      1
1      1
2      1
3      1
4      1
      ..
564    1
565    1
566    1
567    1
568    0
Name: diagnosis, Length: 569, dtype: int32
```

In [51]:
```
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.3,random_state=(
print(x_train.shape)
print(x_test.shape)
print(y_train.shape)
print(y_test.shape)
```

```
(398, 31)
(171, 31)
(398,)
(171,)
```

In [52]:
```
from sklearn.linear_model import LogisticRegression
model=LogisticRegression(penalty = 'l1',C = 1.0,solver = 'liblinear')
model.fit(x_train,y_train)
```

Out[52]:
```
LogisticRegression
LogisticRegression(penalty='l1', solver='liblinear')
```

In [53]:
```
test_prediction=model.predict(x_test)
test_prediction
```

Out[53]:
```
array([1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 1, 1, 1, 1, 1,
       0, 0, 1, 0, 0, 1, 0, 1, 0, 1, 1, 1, 0, 1, 0, 1, 0, 1, 1, 0, 1, 0,
       1, 1, 0, 0, 0, 1, 1, 0, 1, 0, 0, 0, 0, 0, 0, 1, 1, 1, 0, 0, 1, 0,
       1, 1, 1, 0, 1, 1, 0, 0, 1, 0, 0, 0, 0, 0, 1, 1, 1, 0, 1, 0, 0, 0,
       1, 1, 0, 1, 1, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 1, 0,
       0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 1, 0,
       0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 1, 0, 0, 1, 0, 1, 0, 0, 0, 0, 1, 0,
       0, 1, 0, 1, 0, 1, 1, 0, 0, 1, 0, 1, 1, 1, 0, 0, 0])
```

In [38]:
```
train_Acuracy=model.score(x_train,y_train)
print('Train_Accuracy:',train_Acuracy)
test_Acuracy=model.score(x_test,y_test)
print('Test_Accuracy:',test_Acuracy)
```

```
Train_Accuracy: 0.957286432160804
Test_Accuracy: 0.935672514619883
```

In [55]:
```
print(classification_report(y_test,test_prediction))
```

```
              precision    recall  f1-score   support

           0       0.97      0.94      0.96       108
           1       0.91      0.95      0.93        63

    accuracy                           0.95       171
   macro avg       0.94      0.95      0.94       171
weighted avg       0.95      0.95      0.95       171
```

In [56]:
```
from sklearn import svm
svm_model = svm.SVC(C = 1.0,kernel = 'rbf',gamma = 'scale')
svm_model.fit(x_train, y_train)
```

Out[56]:
```
SVC()
```

In [57]:
```
test_prediction=model.predict(x_test)
test_prediction
```

Out[57]:
```
array([1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 1, 1, 1, 1, 1,
       0, 0, 1, 0, 0, 1, 0, 1, 0, 1, 1, 1, 0, 1, 0, 1, 0, 1, 1, 0, 1, 0,
       1, 1, 0, 0, 0, 1, 1, 0, 1, 0, 0, 0, 0, 0, 0, 1, 1, 1, 0, 0, 1, 0,
       1, 1, 1, 0, 1, 1, 0, 0, 1, 0, 0, 0, 0, 0, 1, 1, 1, 0, 1, 0, 0, 0,
       1, 1, 0, 1, 1, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 1, 0,
       0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 1, 0,
       0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 1, 0, 0, 1, 0, 1, 0, 0, 0, 0, 1, 0,
       0, 1, 0, 1, 0, 1, 1, 0, 0, 1, 0, 1, 1, 1, 0, 0, 0])
```

In [58]:
```
train_Acuracy=model.score(x_train,y_train)
print('Train_Accuracy:',train_Acuracy)
test_Acuracy=model.score(x_test,y_test)
print('Test_Accuracy:',test_Acuracy)
```

```
Train_Accuracy: 0.9547738693467337
Test_Accuracy: 0.9473684210526315
```

In [59]:
```
print(classification_report(y_test,test_prediction))
```

```
              precision    recall  f1-score   support

           0       0.97      0.94      0.96       108
           1       0.91      0.95      0.93        63

    accuracy                           0.95       171
   macro avg       0.94      0.95      0.94       171
weighted avg       0.95      0.95      0.95       171
```

In [44]:
```
from sklearn.metrics import accuracy_score
from sklearn.neural_network import MLPClassifier
```

In [45]:
```
from sklearn.preprocessing import StandardScaler
scaler = StandardScaler()
x_train = scaler.fit_transform(x_train)
x_test = scaler.transform(x_test)
```

In [22]:
```
ann_model = MLPClassifier(hidden_layer_sizes=(128),activation='relu',solver='l
ann_model.fit(x_train, y_train)
```

Out[22]:
```
MLPClassifier

MLPClassifier(hidden_layer_sizes=128, solver='lbfgs')
```

In [24]:
```
ann_predictions = ann_model.predict(x_test)
ann_predictions
```

Out[24]:
```
array([1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 1, 1, 1, 1,
       0, 0, 1, 0, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 1, 0, 1, 0,
       0, 1, 0, 0, 0, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 1, 1, 1, 0, 0, 1, 0,
       1, 1, 1, 0, 0, 1, 0, 1, 1, 0, 0, 0, 0, 0, 1, 1, 1, 0, 1, 0, 0, 0,
       1, 1, 0, 1, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 1, 0,
       0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0,
       0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 1, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 1,
       0, 1, 0, 1, 0, 1, 1, 0, 0, 1, 0, 1, 1, 1, 0, 0, 0])
```

In [25]:
```
ann_accuracy = accuracy_score(y_test, ann_predictions)
print("ANN Accuracy:", ann_accuracy)
```

```
ANN Accuracy: 0.9824561403508771
```

In [26]:
```
from sklearn.metrics import classification_report
print(classification_report(y_test,ann_predictions))
```

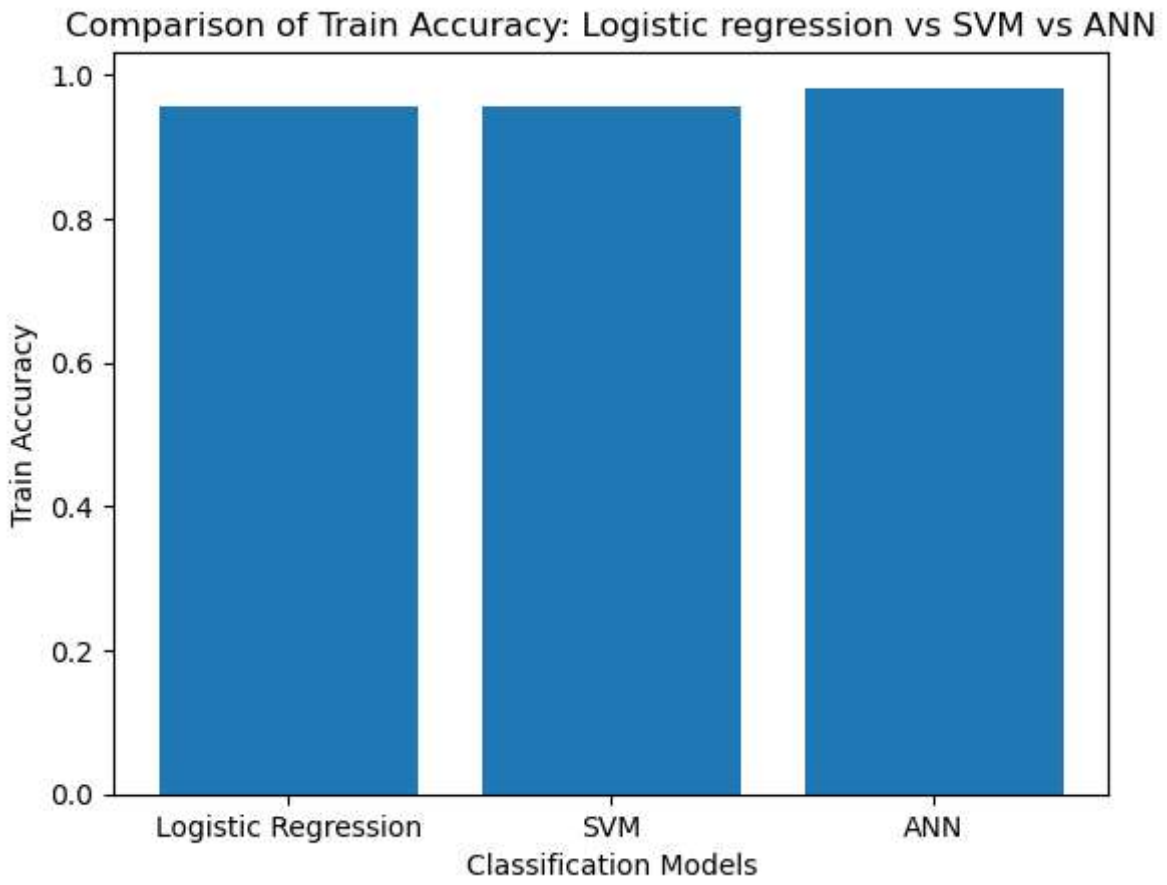
```
              precision    recall  f1-score   support

           0       0.98      0.99      0.99       108
           1       0.98      0.97      0.98        63

    accuracy                           0.98       171
   macro avg       0.98      0.98      0.98       171
weighted avg       0.98      0.98      0.98       171
```

In [29]:
```
import matplotlib.pyplot as plt
x=0.957286432160804
y= 0.957286432160804
z=0.9824561403508771
accuracy_scores = [x,y,z]
model_names = ['Logistic Regression', 'SVM','ANN']
plt.bar(model_names, accuracy_scores)
plt.xlabel('Classification Models')
plt.ylabel('Train Accuracy')
plt.title('Comparison of Train Accuracy: Logistic regression vs SVM vs ANN')

plt.show()
```



In [ ]: