ANN Model:

In [27]:
```
from sklearn.metrics import accuracy_score
from sklearn.neural_network import MLPClassifier
```

In [74]:
```
fromsklearn.preprocessingimportStandardScaler
scaler=StandardScaler()
x_train=scaler.fit_transform(xtrain)
x_test=scaler.fit_transform(xtest)
```

In [75]:
```
ann_model=MLPClassifier(hidden_layer_sizes=(128),activation='relu',solver='lbfgs')
ann_model.fit(x_train,ytrain)
```

Out[75]:
```
MLPClassifier
MLPClassifier(hidden_layer_sizes=128,solver='lbfgs')
```

In [76]:
```
ann_predictions=ann_model.predict(x_test)
ann_predictions
```

Out[76]:
```
array([21, 21,  7,  3,  2,  8, 13,  9, 15,  1, 13,  5, 10, 14, 12,  0,  5,
       10,  5, 12,  4,  2,  9,  8,  6,  5, 10, 16, 13,  9, 19, 20, 11, 15,
        4,  6, 12, 12, 21, 13, 11,  2, 18, 21, 18, 14,  9,  9,  6, 14, 13,
        2,  0, 15, 18,  1, 17, 12, 10,  6, 16, 14, 21, 20, 15,  0,  7,  5,
        0, 16,  4, 19,  9, 11,  7, 13,  3, 11,  8, 12, 20,  2, 21, 21, 15,
        6, 11, 10, 13, 17,  2,  8, 14,  7, 14, 11,  5,  8, 10,  3, 16,  8,
       14,  1,  1, 20, 21,  5, 18, 15, 15, 12,  5,  7, 16, 19, 14, 10, 11,
        8, 19, 10, 16,  3,  3,  2, 19, 16,  3, 17, 13, 13, 15, 14, 11, 14,
        4, 19, 16,  2, 10,  7,  0,  5,  3,  0,  8, 12, 21, 17, 16,  4, 13,
        1, 19,  3, 21,  2,  0,  8, 10, 18,  8,  9,  9, 15, 20, 15,  1, 16,
       18,  0, 13,  4,  6, 14,  9, 19, 17, 16, 20, 17, 17,  9,  9,  1,  4,
       18, 20, 17, 11,  8, 13, 20, 11,  5, 18,  4,  3, 12,  4, 19, 11, 13,
       13, 16, 15, 11, 18,  1,  3,  2, 18, 16, 13, 14, 12, 17, 15, 19,  8,
       20,  2, 17,  2,  5, 11,  5, 16, 20, 13, 14, 16,  9, 19,  4, 12, 14,
        6, 20,  3, 14,  0, 18,  2, 20, 21,  2, 19, 16, 11,  7,  3, 18,  8,
       17, 19,  5, 12, 13,  8, 21, 19, 20,  7,  4,  8, 10,  3,  5,  5, 17,
       19, 11, 20,  3, 18, 16, 19, 18,  4,  9, 19, 15, 13, 12, 10,  1,  2,
       12,  9, 12,  6, 14, 17,  7,  7, 18, 17,  8, 20,  3, 15,  5, 21, 20,
        8, 17,  7, 15,  2, 13, 13,  3,  2, 12,  1, 12, 19,  8, 16, 15,  3,
       10,  6, 17,  7,  9, 10,  0, 20, 15,  0, 17,  2,  8,  3, 13, 10,  7,
        8,  9, 15, 17,  7, 17, 20,  5, 15, 13,  1, 17, 16,  9, 21, 18,  0,
       21, 21, 18,  9, 13,  9,  8,  4,  6,  9, 16,  6, 18, 19,  6,  6,  0,
        6,  0, 16, 11,  7,  1,  0, 13, 20,  9,  1, 20, 10,  3, 19,  1,  3,
       15, 19,  0, 10, 15, 16,  2, 15, 13, 12,  3, 19, 12,  3,  4, 15,  1,
       18, 17,  8, 10,  6, 20,  6,  4, 20,  2, 11, 16, 21, 20,  0,  7, 18,
        7,  3, 12,  8, 19, 11, 12,  7,  1, 14, 18,  1,  6,  2,  0,  0,  8,
        8, 21,  3,  1, 19,  1,  9,  7, 11,  5, 11,  8,  7,  5, 14,  2,  8,
       16, 18, 18, 15, 13, 21, 14, 21, 17, 14, 14, 14, 19, 16, 13,  0,  5,
        4, 11,  4,  7,  7,  3,  3, 12,  9, 17, 16, 14, 17, 18,  2, 17, 15,
        2,  1, 20,  5,  6,  7,  8,  3, 15,  1,  7, 21, 15, 18,  8, 18,  6,
       21, 19,  5,  4, 11, 20, 14,  9, 21, 14,  0,  0, 21,  1, 13, 14,  0,
       14,  6, 20, 17,  6, 17,  3,  0, 19, 13, 20,  2, 12, 16,  8,  1, 17,
        5,  6, 12,  5,  4, 19,  6,  7,  2,  3,  8,  3, 17, 16,  6,  1,  2,
       15, 17,  0, 16, 19, 11, 18, 17, 12, 19, 17,  7, 20,  6,  8, 13, 10,
       13,  9,  1, 13,  0, 13, 15,  4,  3, 10,  9, 13,  7,  7, 16, 20,  2,
        1,  6, 11, 10, 20, 20,  4, 13,  6,  5, 17,  5, 14, 14, 16, 19,  3,
       10, 11, 12, 16,  5, 20, 17, 17,  4, 20,  6, 13,  4, 20,  7,  0,  1,
        4,  1, 11, 12, 17, 17, 20, 17, 15,  6, 10,  9,  2,  5, 20, 16,  4,
        1,  2,  0, 11, 12,  3,  4, 15,  5, 19, 16,  7, 17,  3, 20, 21, 16,
        9, 16, 16, 10, 11, 12,  9, 19,  4, 13, 11, 10, 14, 17,  9, 16, 10,
        5, 14, 15,  4,  7,  4, 19, 18, 17, 10, 17,  1,  3, 13, 17, 16, 13,
       19,  2, 20, 16, 20,  3,  2, 18,  5,  3,  7,  4,  3,  7,  5, 19, 19,
        1,  3,  2, 18, 13,  0, 19,  0,  2,  0, 21,  9])
```

In [77]:
```
ann_accuracy=accuracy_score(ytest,ann_predictions) print("ANN
Accuracy:", ann_accuracy)
```

```
ANNAccuracy:0.9862258953168044
```

In [78]:
```
train_Acuracy=ann_model.score(x_train,ytrain)
print('Train_Accuracy:',train_Acuracy)
test_Acuracy=ann_model.score(x_test,ytest)
print('Test_Accuracy:',test_Acuracy)
```

```
Train_Accuracy:1.0
Test_Accuracy:0.9862258953168044
```

In [79]:
```
from sklearn.metrics import classification_report
print(classification_report(ytest,ann_predictions))
```

```
              precision    recall  f1-score   support

           0       1.00      1.00      1.00        31
           1       1.00      0.97      0.98        32
           2       0.97      0.94      0.96        35
           3       1.00      1.00      1.00        39
           4       1.00      1.00      1.00        30
           5       1.00      1.00      1.00        32
           6       0.97      1.00      0.98        28
           7       1.00      1.00      1.00        33
           8       0.94      0.97      0.95        31
           9       0.97      1.00      0.98        30
          10       0.93      0.96      0.94        26
          11       1.00      1.00      1.00        29
          12       1.00      1.00      1.00        29
          13       0.97      1.00      0.99        39
          14       1.00      1.00      1.00        31
          15       0.97      1.00      0.98        31
          16       1.00      1.00      1.00        40
          17       1.00      1.00      1.00        43
          18       1.00      0.94      0.97        32
          19       1.00      1.00      1.00        38
          20       0.97      0.95      0.96        40
          21       1.00      0.96      0.98        27

    accuracy                           0.99       726
   macroavg       0.99      0.99      0.99       726
weightedavg       0.99      0.99      0.99       726
```

Comparision between Logistic Regresion, RandomForestRegression and ANN_Model using BarGraph:

In [80]:
```
logistic_regression_accuracy=0.9435261707988981
random_forest_accuracy=0.9715513770838728
ann_model=0.9862258953168044
accuracy_scores=[logistic_regression_accuracy,random_forest_accuracy,ann_model]
model_names=['LogisticRegression','RandomForestRegression','ANN_Model']
plt.bar(model_names,accuracy_scores)
plt.xlabel('Models')
plt.ylabel('Test_Accuracy')
plt.title('ComparisionofTest_Accuarcy:LogisticRegressionv/sRandomForestRegressionv/sANN
plt.show()
```

Result:

In this case, x-axis represents the all three models and y-axis represents the test_accuracy.It is comparison between Logistic Regression ,Random Forest Regression and ANN_Model.

Logistic Regression Accuracy: 0.94
Random Forest Accuracy: 0.97
ANN_Model Accuracy: 0.98

The ANN_Model has given the best accuracy among three models.