Date: **Ex No:** 8	**Title of the Lab** Implementation of a Supervised Machine Learning algorithms for Telco Churn Analysis Dataset	**Name:** Yuvraj Singh Chauhan **Registration Number:** RA1911027010058 **Section:** N1 **Lab Batch:** 1 **Day Order:** 3

Aim:

To find the best fitting algorithm for Telco Churn Analysis dataset.

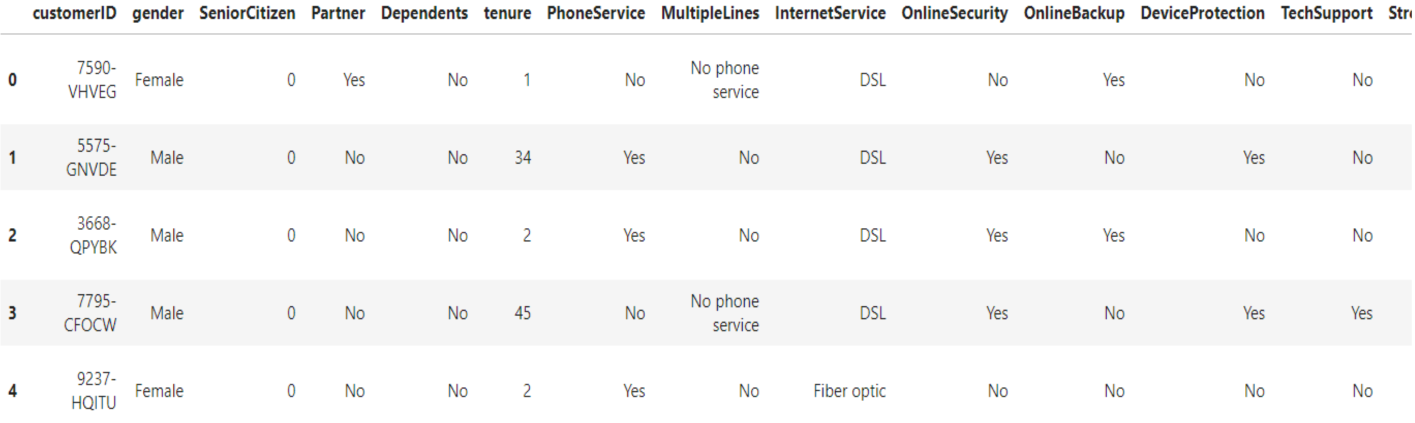
Description of Telco Churn Analysis:

For Telco companies it is key to attract new customers and at the same time avoid contract terminations (=churn) to grow their revenue generating base. Looking at churn, different reasons trigger customers to terminate their contracts, for example better price offers, more interesting packages, bad service experiences or change of customers' personal situations.

Churn analytics provides valuable capabilities to predict customer churn and also define the underlying reasons that drive it. The churn metric is mostly shown as the percentage of customers that cancel a product or service within a given period (mostly months). If a Telco company had 10 Mio. customers on the 1st of January and received 500K contract terminations until the 31st of January the monthly churn for January would be 5%.

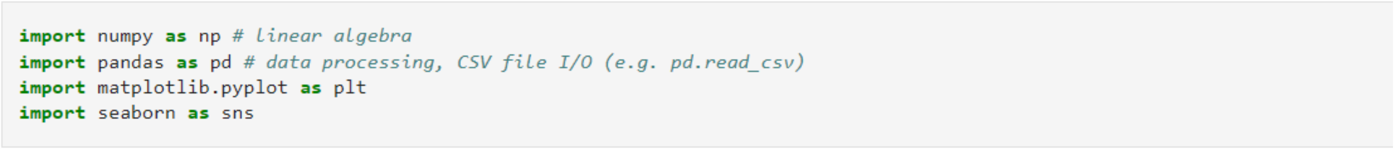
Telcos apply machine learning models to predict churn on an individual customer basis and take counter measures such as discounts, special offers or other gratifications to keep their customers. A customer churn analysis is a typical classification problem within the domain of supervised learning.

Dataset:

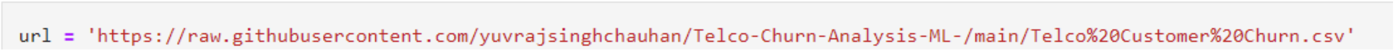


	customerID	gender	SeniorCitizen	Partner	Dependents	tenure	PhoneService	MultipleLines	InternetService	OnlineSecurity	OnlineBackup	DeviceProtection	TechSupport	Str
0	7590-VHVEG	Female	0	Yes	No	1	No	No phone service	DSL	No	Yes	No	No	
1	5575-GNVDE	Male	0	No	No	34	Yes	No	DSL	Yes	No	Yes	No	
2	3668-QPYBK	Male	0	No	No	2	Yes	No	DSL	Yes	Yes	No	No	
3	7795-CFOCW	Male	0	No	No	45	No	No phone service	DSL	Yes	No	Yes	Yes	
4	9237-HQITU	Female	0	No	No	2	Yes	No	Fiber optic	No	No	No	No	

Importing Files for ML:

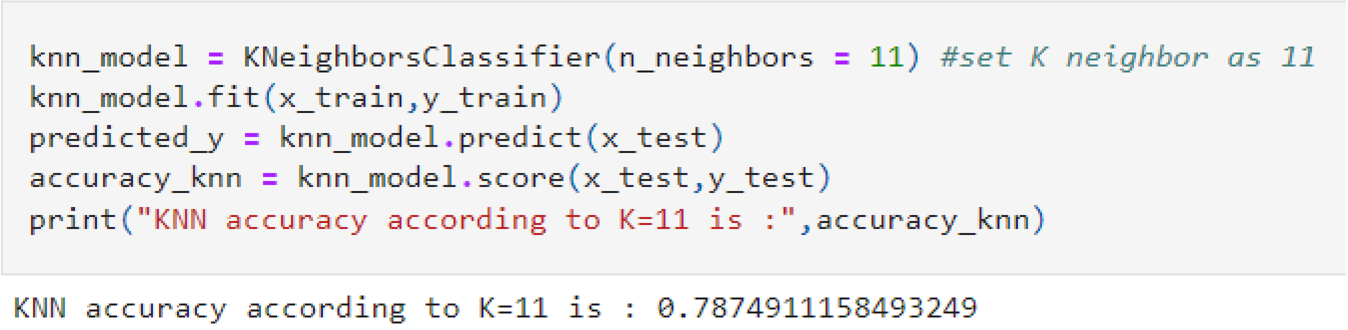


```
import numpy as np # linear algebra
import pandas as pd # data processing, CSV file I/O (e.g. pd.read_csv)
import matplotlib.pyplot as plt
import seaborn as sns
```



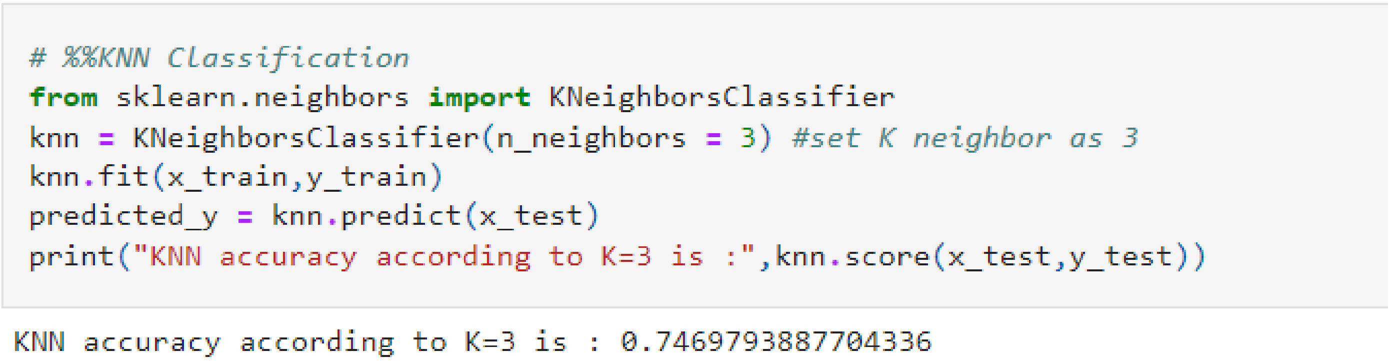
```
url = 'https://raw.githubusercontent.com/yuvrajsinghchauhan/Telco-Churn-Analysis-ML-/main/Telco%20Customer%20Churn.csv'
```

KNN Classifier:



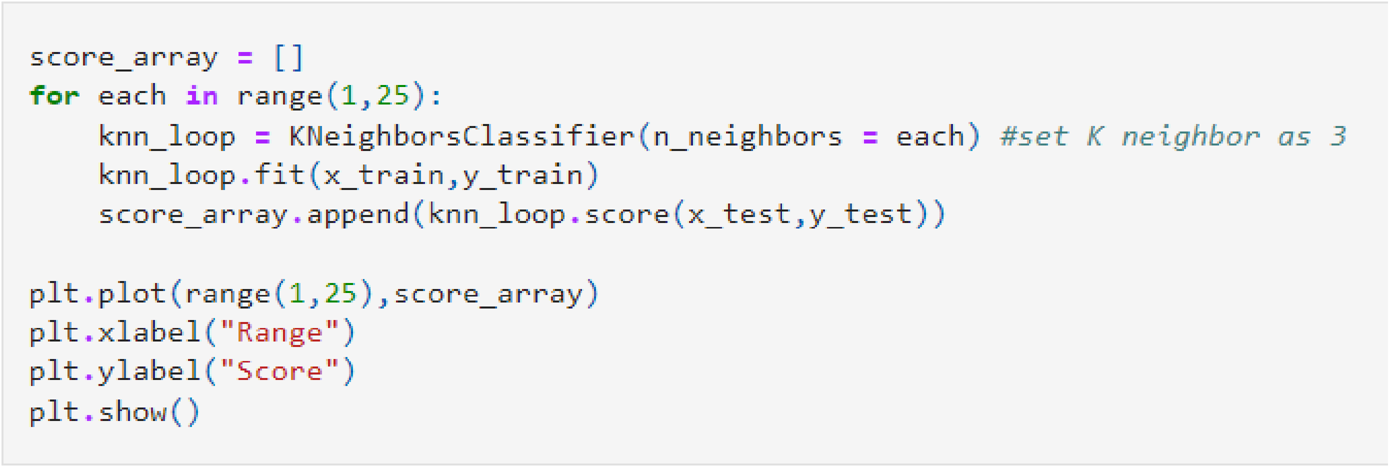
```
knn_model = KNeighborsClassifier(n_neighbors = 11) #set K neighbor as 11
knn_model.fit(x_train,y_train)
predicted_y = knn_model.predict(x_test)
accuracy_knn = knn_model.score(x_test,y_test)
print("KNN accuracy according to K=11 is :",accuracy_knn)
```

```
KNN accuracy according to K=11 is : 0.7874911158493249
```



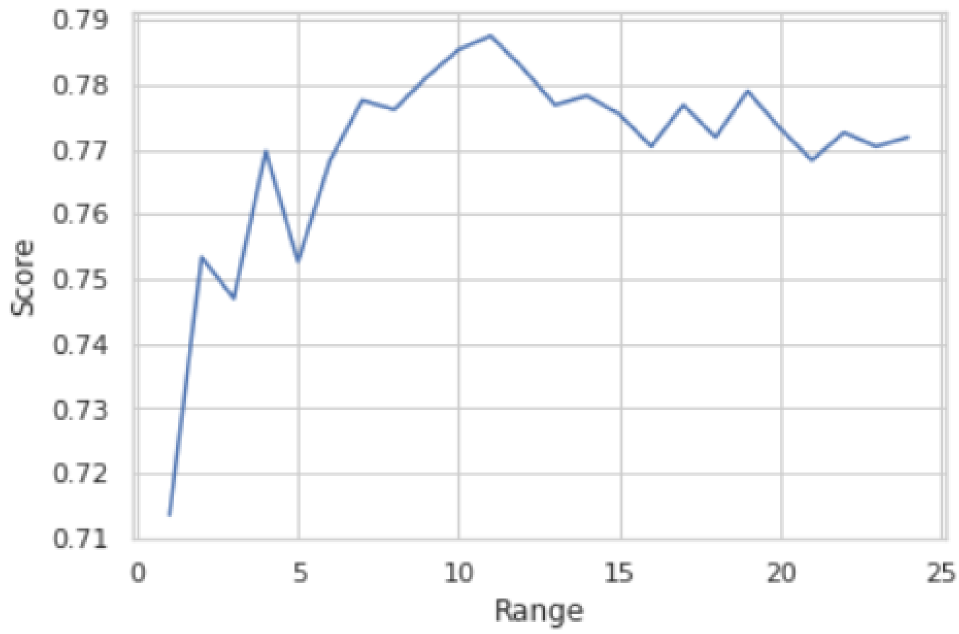
```
# %%KNN Classification
from sklearn.neighbors import KNeighborsClassifier
knn = KNeighborsClassifier(n_neighbors = 3) #set K neighbor as 3
knn.fit(x_train,y_train)
predicted_y = knn.predict(x_test)
print("KNN accuracy according to K=3 is :",knn.score(x_test,y_test))
```

```
KNN accuracy according to K=3 is : 0.7469793887704336
```

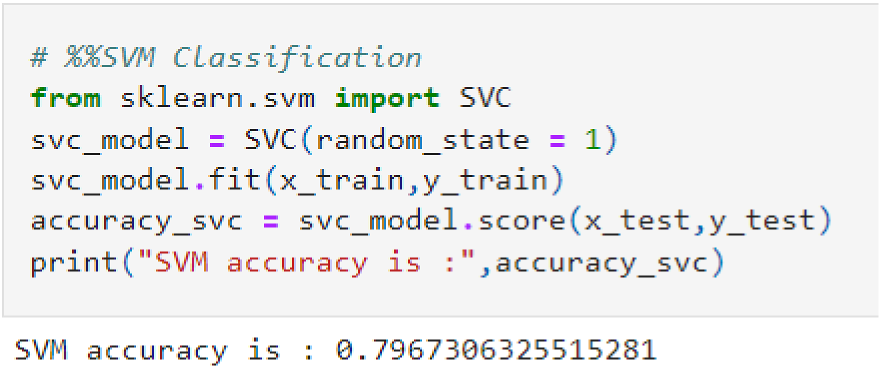


```
score_array = []
for each in range(1,25):
    knn_loop = KNeighborsClassifier(n_neighbors = each) #set K neighbor as 3
    knn_loop.fit(x_train,y_train)
    score_array.append(knn_loop.score(x_test,y_test))

plt.plot(range(1,25),score_array)
plt.xlabel("Range")
plt.ylabel("Score")
plt.show()
```



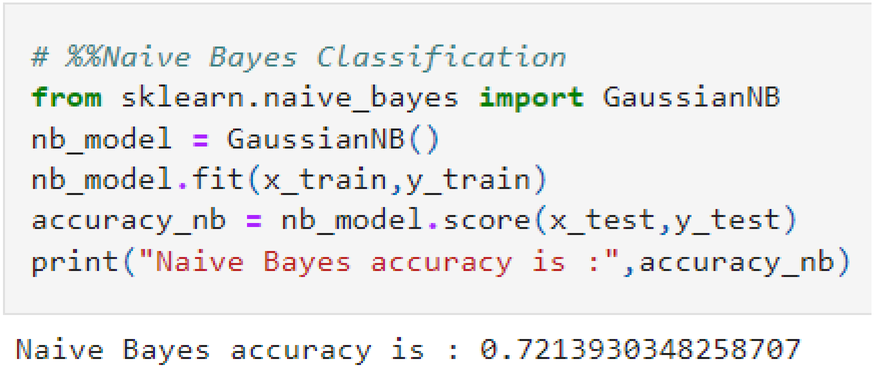
Support Vector Machine:



```
# %%SVM Classification
from sklearn.svm import SVC
svc_model = SVC(random_state = 1)
svc_model.fit(x_train,y_train)
accuracy_svc = svc_model.score(x_test,y_test)
print("SVM accuracy is :",accuracy_svc)
```

```
SVM accuracy is : 0.7967306325515281
```

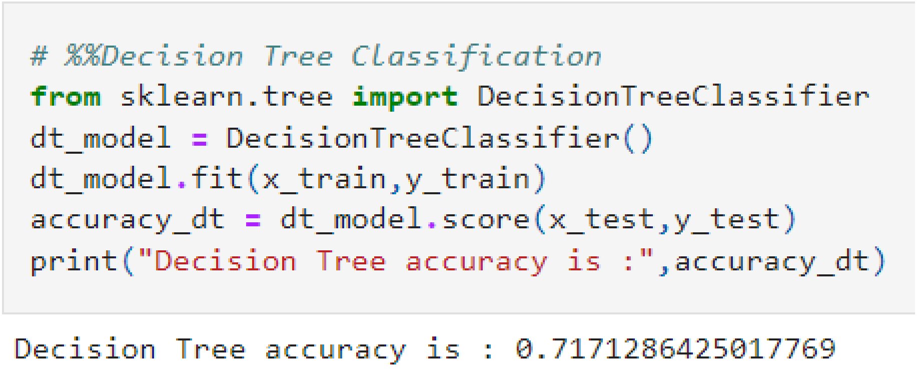
Naïve Bayes Classification:



```
# %%Naive Bayes Classification
from sklearn.naive_bayes import GaussianNB
nb_model = GaussianNB()
nb_model.fit(x_train,y_train)
accuracy_nb = nb_model.score(x_test,y_test)
print("Naive Bayes accuracy is :",accuracy_nb)
```

```
Naive Bayes accuracy is : 0.7213930348258707
```

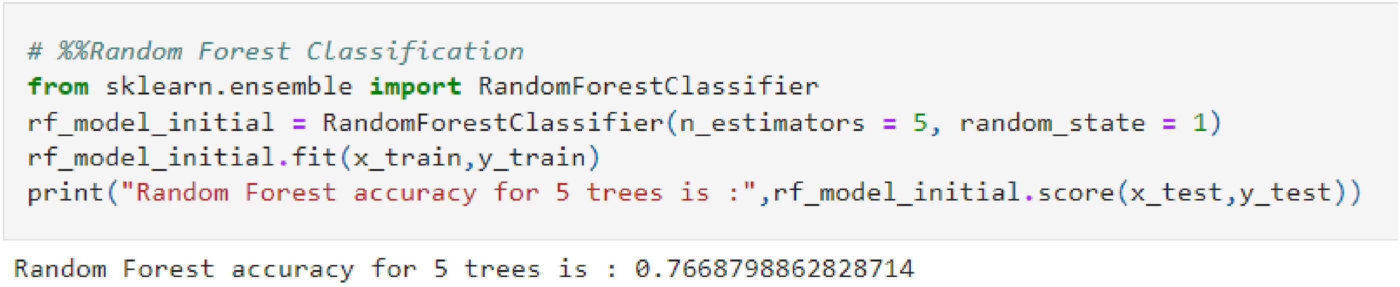
Decision Tree Classification:



```
# %%Decision Tree Classification
from sklearn.tree import DecisionTreeClassifier
dt_model = DecisionTreeClassifier()
dt_model.fit(x_train,y_train)
accuracy_dt = dt_model.score(x_test,y_test)
print("Decision Tree accuracy is :",accuracy_dt)
```

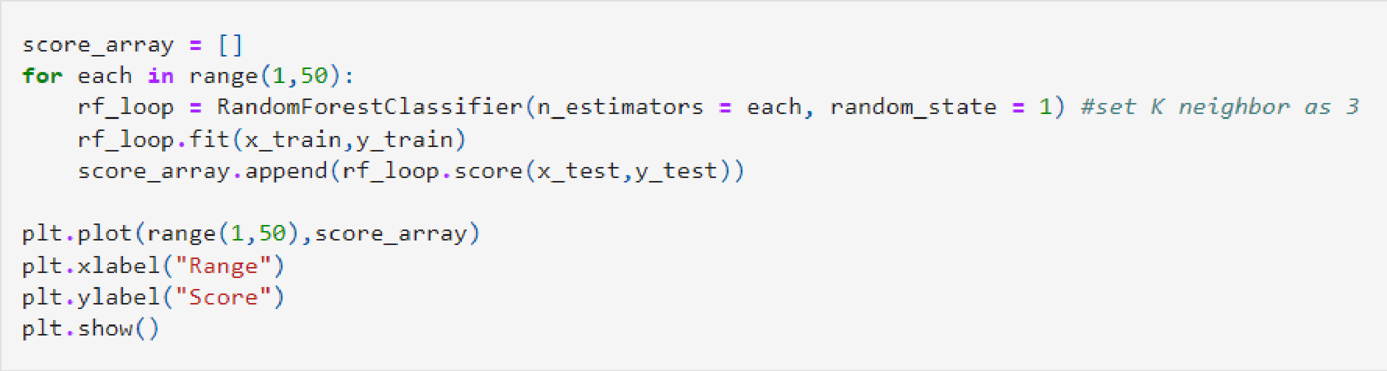
```
Decision Tree accuracy is : 0.7171286425017769
```

Random Forest Classification:



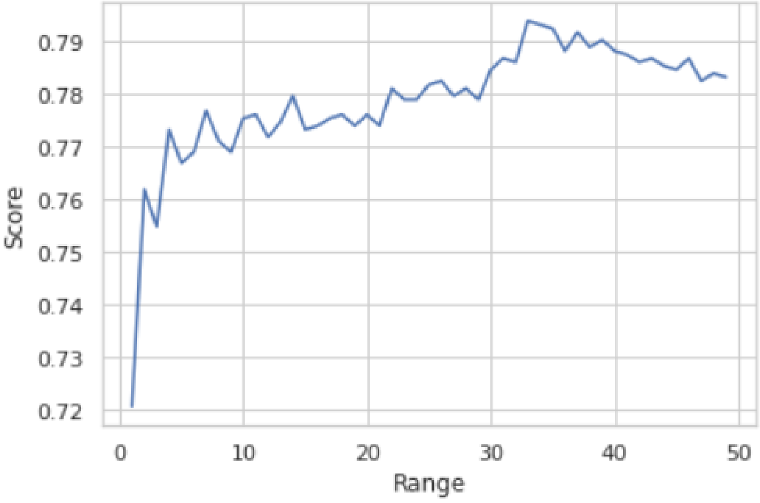
```
# %%Random Forest Classification
from sklearn.ensemble import RandomForestClassifier
rf_model_initial = RandomForestClassifier(n_estimators = 5, random_state = 1)
rf_model_initial.fit(x_train,y_train)
print("Random Forest accuracy for 5 trees is :",rf_model_initial.score(x_test,y_test))
```

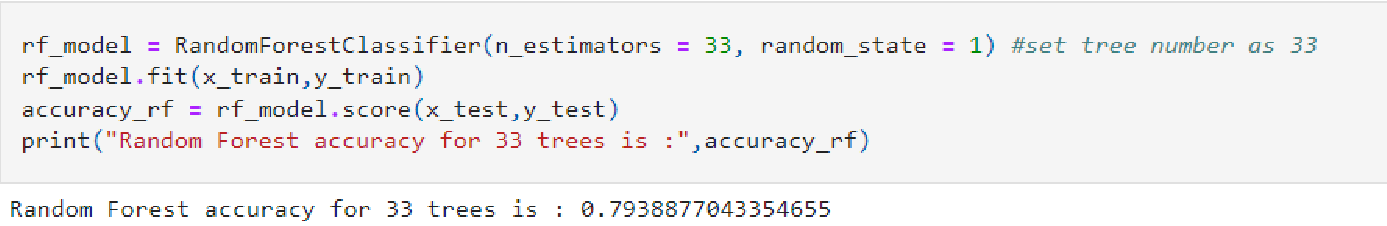
```
Random Forest accuracy for 5 trees is : 0.7668798862828714
```



```
score_array = []
for each in range(1,50):
    rf_loop = RandomForestClassifier(n_estimators = each, random_state = 1) #set K neighbor as 3
    rf_loop.fit(x_train,y_train)
    score_array.append(rf_loop.score(x_test,y_test))

plt.plot(range(1,50),score_array)
plt.xlabel("Range")
plt.ylabel("Score")
plt.show()
```





```
rf_model = RandomForestClassifier(n_estimators = 33, random_state = 1) #set tree number as 33
rf_model.fit(x_train,y_train)
accuracy_rf = rf_model.score(x_test,y_test)
print("Random Forest accuracy for 33 trees is :",accuracy_rf)
```

```
Random Forest accuracy for 33 trees is : 0.7938877043354655
```

Conclusion:

So, from the above comparison it can be observed that the model accuracy for the SVM and Random Forest Classifier are the highest (i.e., 79%), however higher accuracy doesn’t mean that they will be the best always as depending on the model and data it can either overfit or take very long time to give the output and it could be more prone to outliers as well.

Therefore, taking all those things into consideration the best model for our dataset would be “Support Vector Machine”.

Signature of the Student

[YUVRAJ SINGH CHAUHAN]