DEPARTMENT OF COMPUTER & SOFTWARE ENGINEERING

COLLEGE OF E&ME, NUST, RAWALPINDI

EC452 Machine Learning

Feature Engineering, Selection and Dimensionality Reduction

SUBMITTED TO:

Prof Dr Asad Mansoor khan

SUBMITTED BY:

Uzair Waheed

Reg # 371828

DE- 43 CE

Submission Date:22/02/24

Feature Engineering, Selection and Dimensionality Reduction

Tasks:

1. Apply k-Nearest Neighbor (kNN), for Task 2, Task 3, and Task 4: a. Experiment with varying value of k, plot accuracy against different k values and highlight the best value of k for this problem.

2. Apply feature selection to reduce the dimensions of data. Discuss your results explaining why you have selected the features that you selected and why you dropped others.

3. Apply Principal Component Analysis to reduce the dimensions of the data. Select an appropriate number of PCA dimensions. Discuss your results explaining why you have kept this specific number of PCA dimensions.

4. Apply Linear Discriminant Analysis to reduce dimensions. Discuss your results.

5. Report performance of kNN in terms of accuracy.

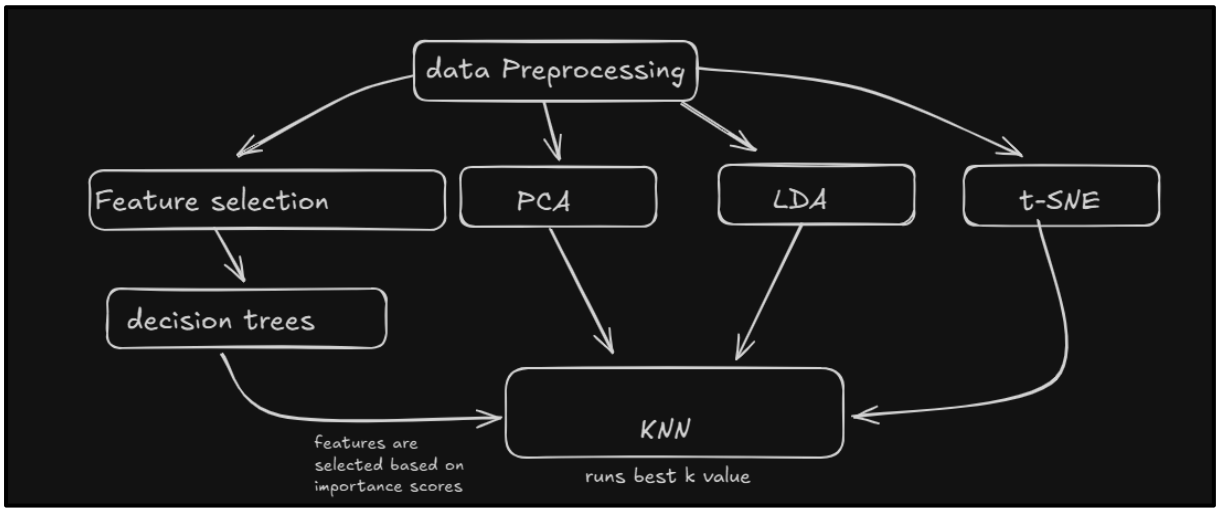
6. Randomize all samples and divide the dataset as class-wise 70:30 for training and testing. Repeat your experiments for 5 times and report average along with standard deviation of your results.

7. Compare all results and discuss the findings.

Solution:

For every feature selection/dimension reduction method we calculate knn for k (1->21) then select best k on basis of accuracy score. For feature selection we use Embedded method in which we use decision tree to know importance of features. We selected top-k features and test which are best for us. For PCA we calculate from 1-10 dimensions and then select how many dimensions are good for us. Similarly we have applied other dimension reduction tecnhiques like LDA and t-SNE which is visualization technique used to reduce multi-dimensional data to 2/3 D format not effective as LDA or PCA but good for visualization of data.

Flowchart:



Code:

```
import pandas as pd
from sklearn.preprocessing import StandardScaler
from sklearn.model_selection import train_test_split
from sklearn.neighbors import KNeighborsClassifier
from sklearn.preprocessing import LabelEncoder
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import accuracy_score
import matplotlib.pyplot as plt
from sklearn.decomposition import PCA
from sklearn.discriminant_analysis import LinearDiscriminantAnalysis
from sklearn.manifold import TSNE


def preprocessing(dataset):
    dataset.drop(columns=["id"], inplace=True)
    dataset["bmi"].fillna(dataset["bmi"].median(), inplace=True)
    categorical_columns = ["gender", "ever_married", "work_type",
"Residence_type", "smoking_status"]
    label_encoders = {}
    for col in categorical_columns:
        le = LabelEncoder()
        dataset[col] = le.fit_transform(dataset[col])
        label_encoders[col] = le
    return dataset

def KNN(features,out):
    f_train, f_test, o_train, o_test = train_test_split(features, out,
test_size=0.3, stratify=out)
    acc_arr =[]
    k=[]
    for i in range(23):
```

```
        if(i%2 != 0):
            knn = KNeighborsClassifier(n_neighbors=i)
            knn.fit(f_train, o_train)
            o_pred = knn.predict(f_test)
            acc=accuracy_score(o_test, o_pred)
            print(f"K={i} ",acc)
            acc_arr.append(acc)

    m = max(acc_arr)
    ind = acc_arr.index(m)
    k_value = 2*ind +1
    k= list(range(1, 23, 2))
    plt.xticks(k)
    plt.scatter(k,acc_arr,color='b')
    plt.xlabel("K-value")
    plt.ylabel("Accuracy")
    plt.title("k-values VS Accuracy")
    plt.show()
    print("Best K-value is ",k_value)
    return m

def plot(feature_df):
    plt.figure(figsize=(10, 5))
    plt.barh(feature_df["Feature"], feature_df["Importance"], color="skyblue")
    plt.xlabel("Feature Importance")
    plt.ylabel("Feature Name")
    plt.title("Feature Importance from Decision Tree")
    plt.show()


def feature_selection(feature,output,dataset):
    print("FEATURE SELECTION: ")
    #using decision tree to check the importance of features
    f_train, f_test, o_train, o_test = train_test_split(feature, output,
test_size=0.3, stratify=output)
    acc10 = KNN(feature,output)
    print("Accurracy with 10-f",acc10)
    res = DecisionTreeClassifier()
    res.fit(f_train, o_train)
    feature_importances = res.feature_importances_
    feature_df = pd.DataFrame({"Feature": f_train.columns, "Importance":
feature_importances})
    sorted_values = feature_df.sort_values(by="Importance", ascending=False)
    print(sorted_values)
    plot(sorted_values)

    features=dataset.drop(columns=[sorted_values.Feature[9],sorted_values.Feat
ure[8],sorted_values.Feature[7]])
    out = dataset["stroke"]
```

```
    acc = KNN(features,out)
    print("Accurracy with 7-f",acc)

    features=dataset.drop(columns=[sorted_values.Feature[9],sorted_values.Feat
ure[8],sorted_values.Feature[7],
    sorted_values.Feature[6],sorted_values.Feature[5]])
    out = dataset["stroke"]
    acc = KNN(features,out)
    print("Accurracy with 5-f",acc)

    features=dataset[[sorted_values.Feature[0], sorted_values.Feature[1],
sorted_values.Feature[2]]]
    out = dataset["stroke"]
    acc = KNN(features,out)
    print("Accurracy with 3-f",acc)


def Pca(feature,output):
    print("PCA : ")
    acc_pca = []
    for i in range(1,11):
        pca = PCA(n_components=i)
        pca_features = pca.fit_transform(feature)
        acc = KNN(pca_features,output)
        print(f"Accuracy with PCA{i}:",acc)
        acc_pca.append(acc)
    m = max(acc_pca)
    ind = acc_pca.index(m)
    dim = ind + 1  #where return tuple use index 0 for array
    print("Selected-PCA Dimensions are ",dim)
    return m

def Lda(feature,output):
    print("LDA:")
    lda = LinearDiscriminantAnalysis(n_components=1)
    lda_features = lda.fit_transform(feature,output)
    acc = KNN(lda_features,output)
    print("Accuracy with LDA:",acc)

def tSNE(feature,output):
    print("t_SNE: ")
    tsne = TSNE(n_components=2, perplexity=30)
    tsne_feature = tsne.fit_transform(feature)
    acc = KNN(tsne_feature,output)
    print("Accuracy with tSNE: ",acc)

#main
dataset = pd.read_csv("healthcare-dataset-stroke-data.csv")
dataset = preprocessing(dataset)
```

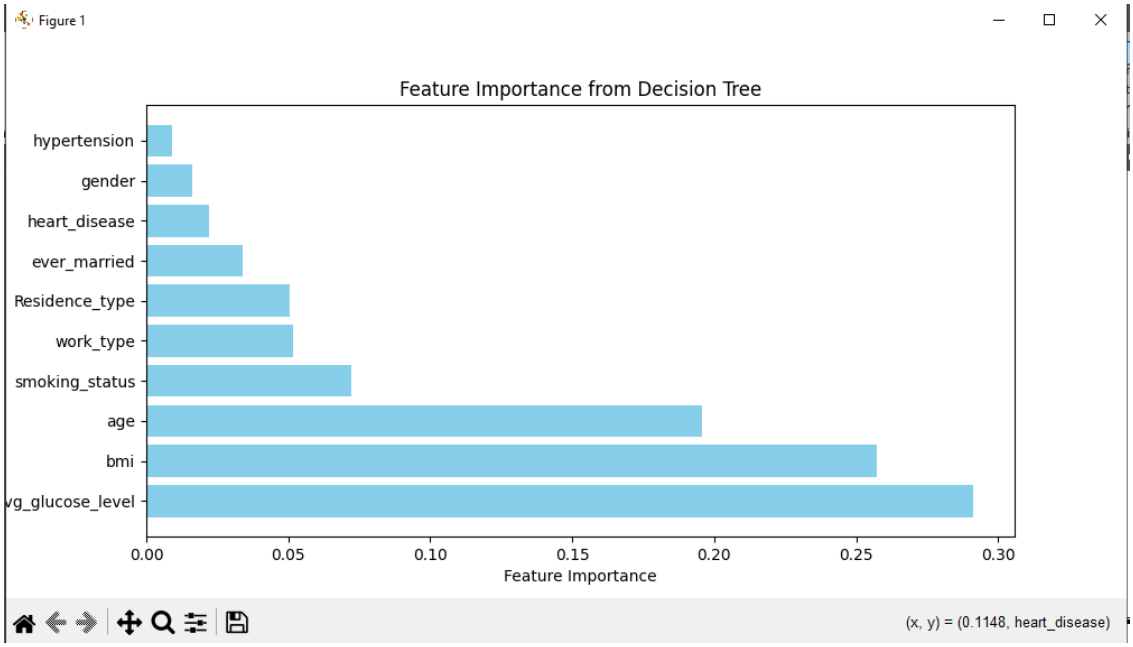
```
feature = dataset.drop(columns=["stroke"])
output = dataset["stroke"]
feature_selection(feature,output,dataset)
Pca(feature,output)
Lda(feature,output)
tSNE(feature,output)
# print(dataset)
```

Output:

Feature Selection:

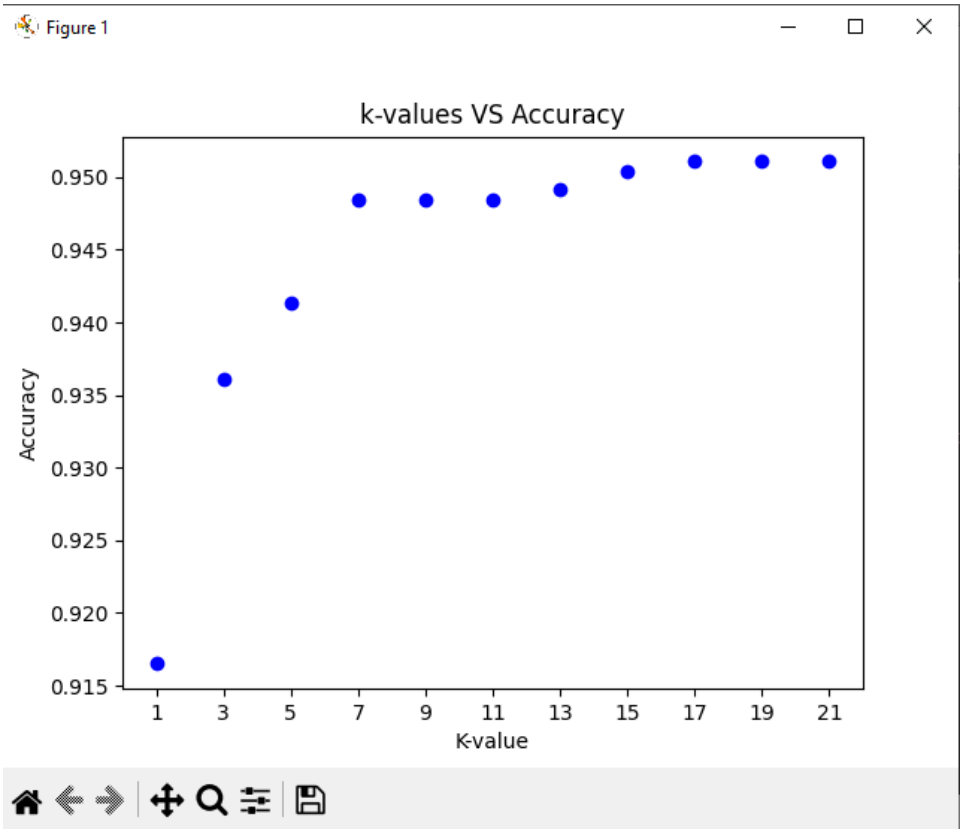
Importance of features calculated using decision tree.

	Feature	Importance
7	avg_glucose_level	0.291204
8	bmi	0.257199
1	age	0.195700
9	smoking_status	0.072350
5	work_type	0.051811
6	Residence_type	0.050672
4	ever_married	0.034050
3	heart_disease	0.022000
0	gender	0.016131
2	hypertension	0.008884

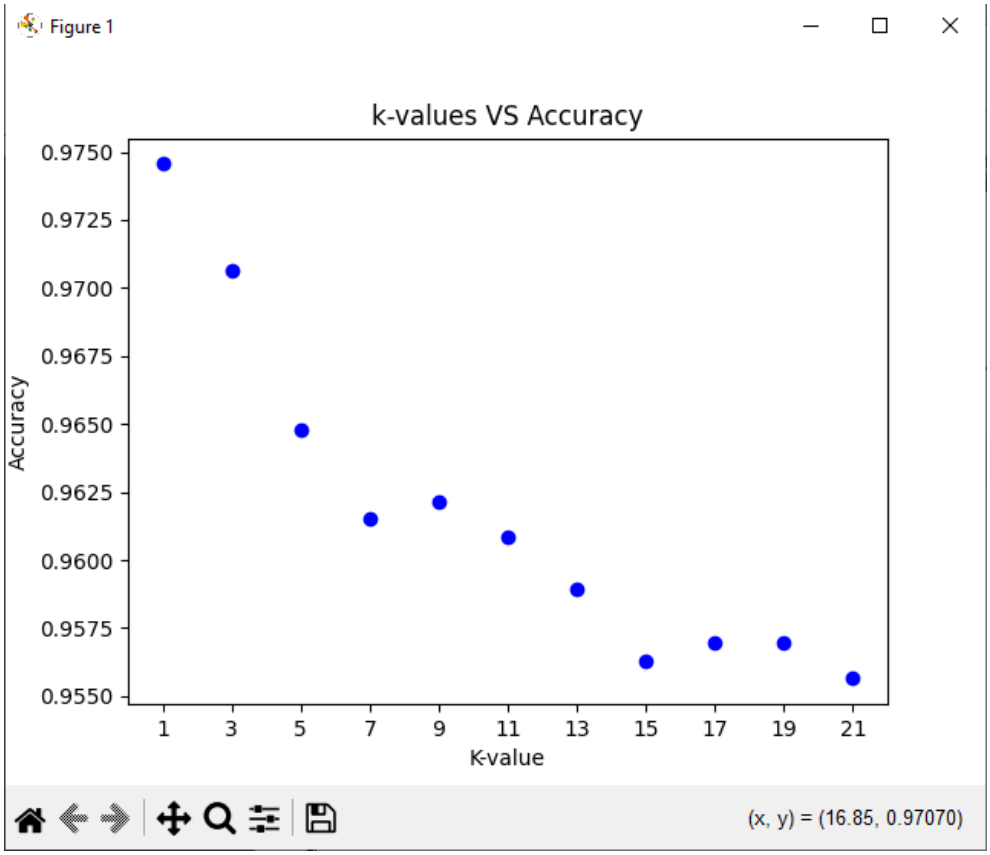


When we selected all features

```
FEATURE SELECTION:
K=1  0.9165035877364645
K=3  0.9360730593607306
K=5  0.9412915851272016
K=7  0.9484670580560991
K=9  0.9484670580560991
K=11  0.9484670580560991
K=13  0.949119373776908
K=15  0.9504240052185258
K=17  0.9510763209393346
K=19  0.9510763209393346
K=21  0.9510763209393346
Best K-value is  17
Accurracy with 10-f 0.9510763209393346
```

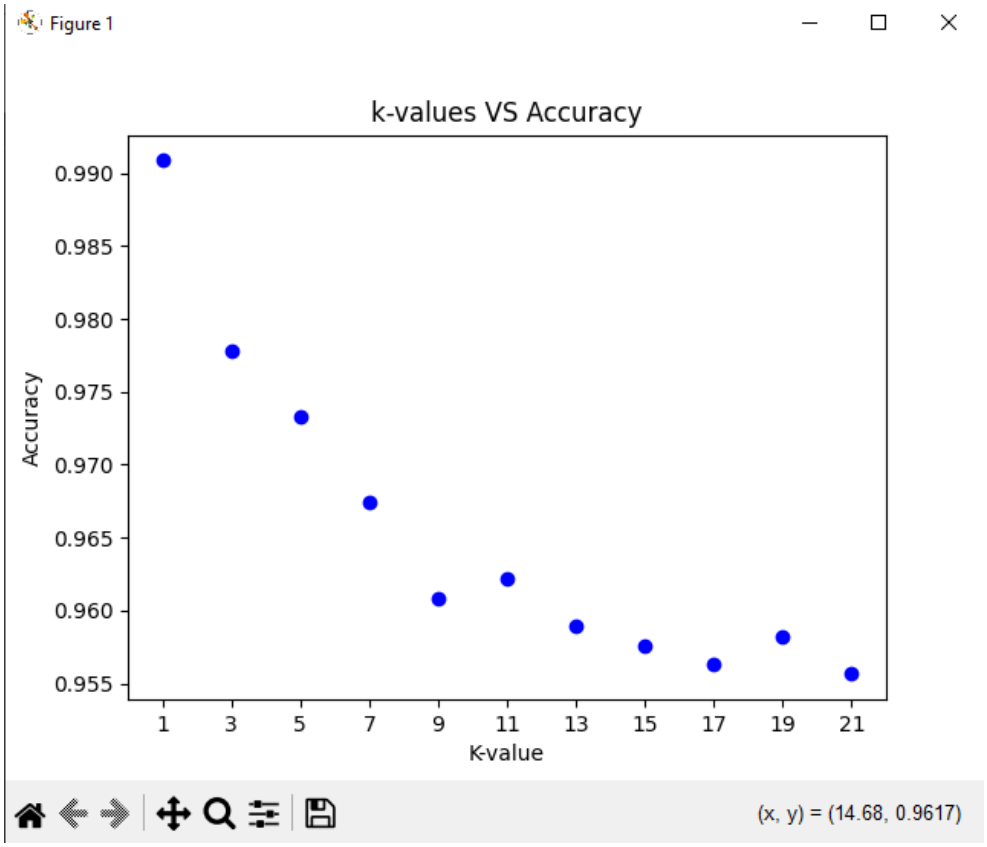


When we selected top 7 features



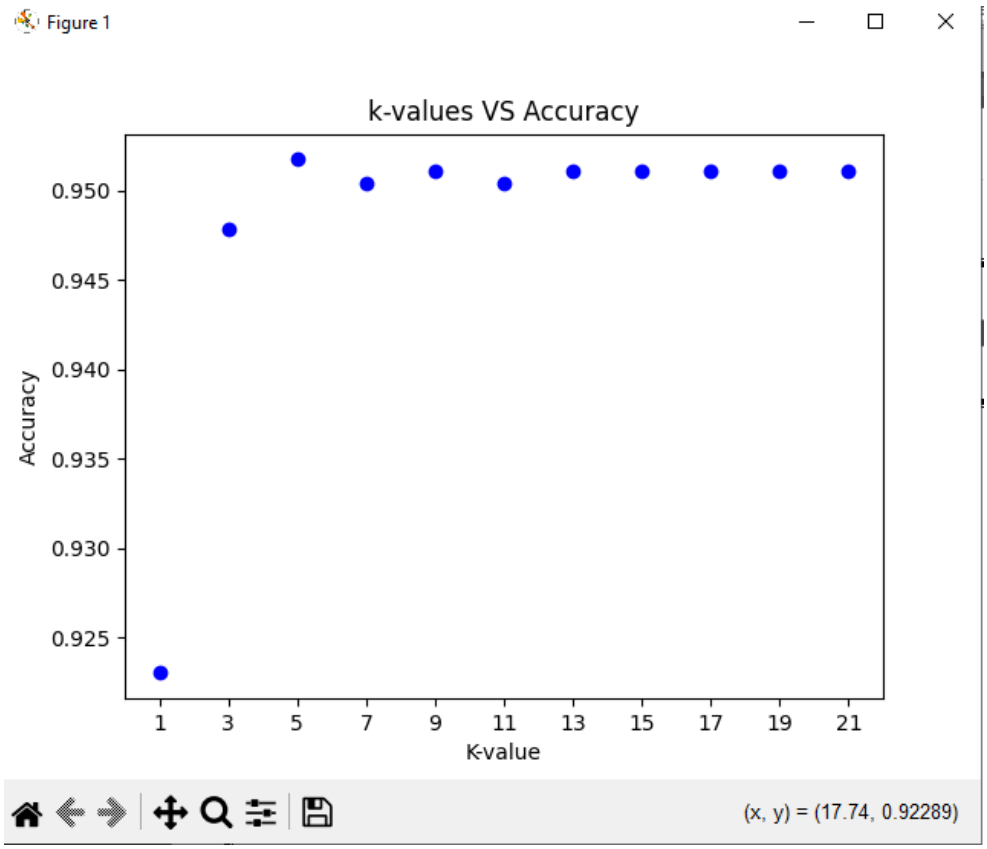

```
K=1  0.974559686888454
K=3  0.9706457925636007
K=5  0.9647749510763209
K=7  0.9615133724722765
K=9  0.9621656881930855
K=11  0.9608610567514677
K=13  0.958904109589041
K=15  0.9562948467058056
K=17  0.9569471624266145
K=19  0.9569471624266145
K=21  0.9556425309849967
Best K-value is  1
Accurracy with 7-f 0.974559686888454
```

When we selected top 5 features




```
K=1  0.9908675799086758
K=3  0.9778212654924984
K=5  0.9732550554468362
K=7  0.9673842139595564
K=9  0.9608610567514677
K=11  0.9621656881930855
K=13  0.958904109589041
K=15  0.9575994781474233
K=17  0.9562948467058056
K=19  0.9582517938682322
K=21  0.9556425309849967
Best K-value is  1
Accurracy with 5-f 0.9908675799086758
```

When we selected top 3 features

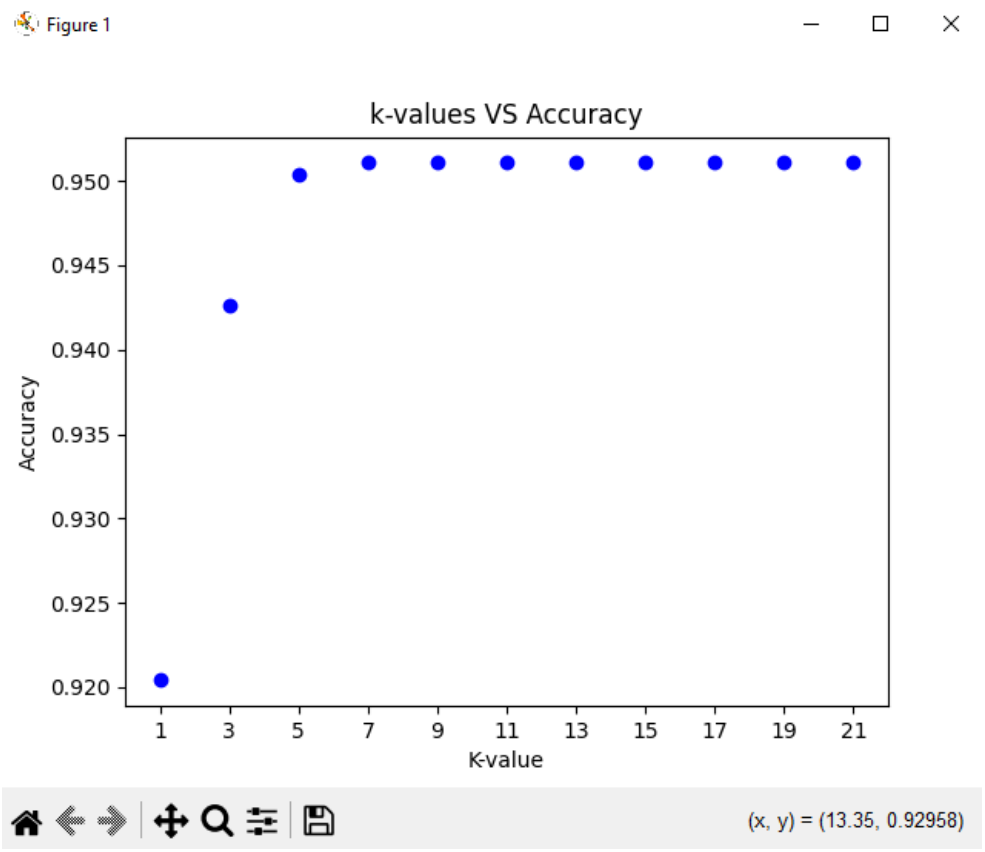



```
K=1  0.9230267449445532
K=3  0.9478147423352903
K=5  0.9517286366601435
K=7  0.9504240052185258
K=9  0.9510763209393346
K=11  0.9504240052185258
K=13  0.9510763209393346
K=15  0.9510763209393346
K=17  0.9510763209393346
K=19  0.9510763209393346
K=21  0.9510763209393346
Best K-value is  5
Accurracy with 3-f 0.9517286366601435
```

Accuracy dropped from 0.99 to 0.95 when we selected top 3 features instead of top 5, so top 5 features should be selected for best accuracy.
Note: we are sorting feature on importance scores given by decision tree.

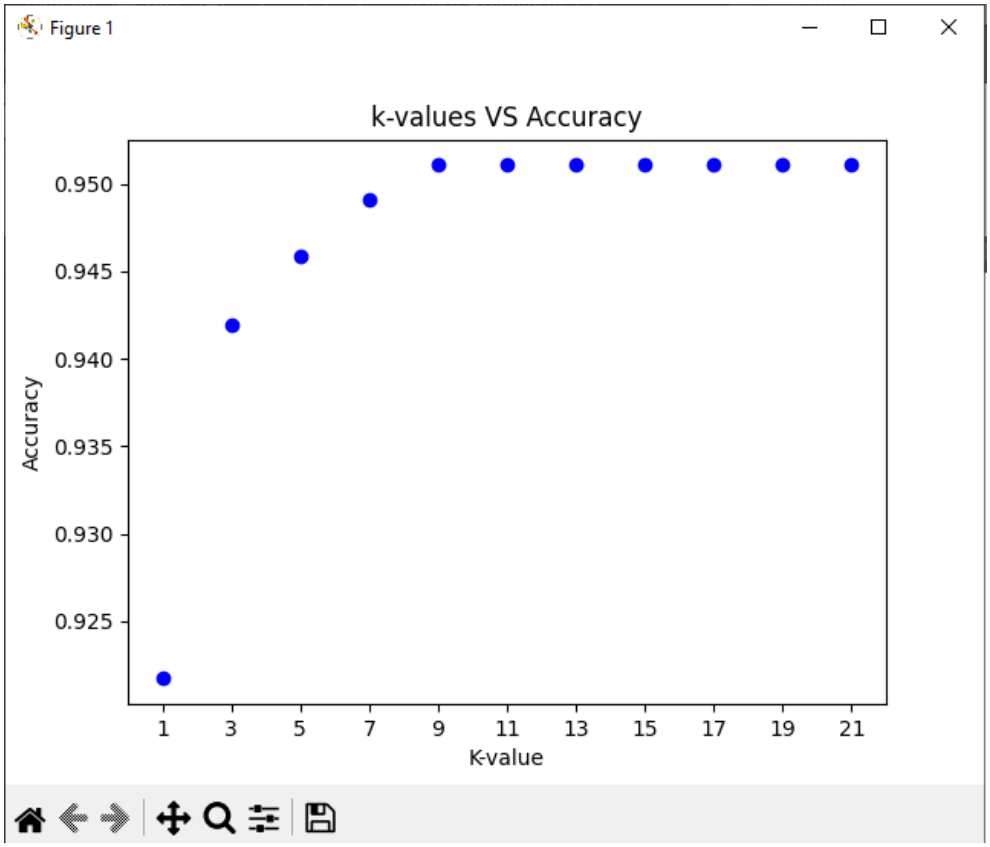
PCA:

With 1 dimension.



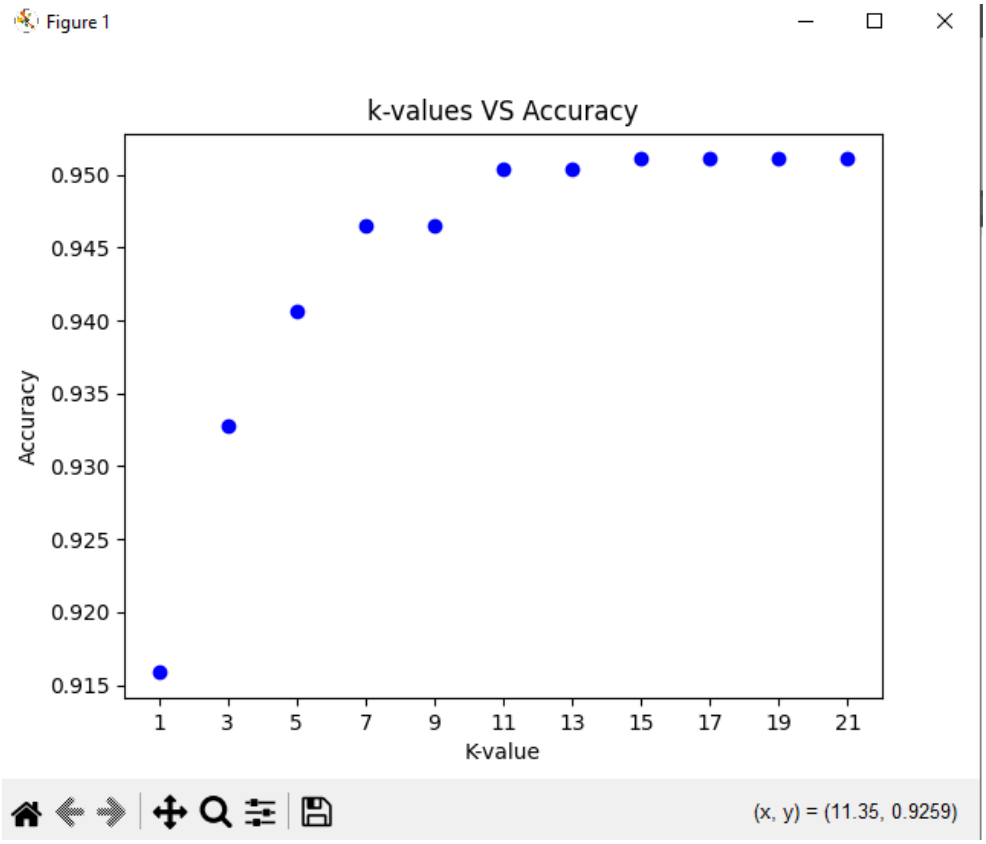

```
K=1  0.9204174820613177
K=3  0.9425962165688193
K=5  0.9504240052185258
K=7  0.9510763209393346
K=9  0.9510763209393346
K=11  0.9510763209393346
K=13  0.9510763209393346
K=15  0.9510763209393346
K=17  0.9510763209393346
K=19  0.9510763209393346
K=21  0.9510763209393346
Best K-value is  7
Accuracy with PCA1: 0.9510763209393346
```

With 2 dimensions.



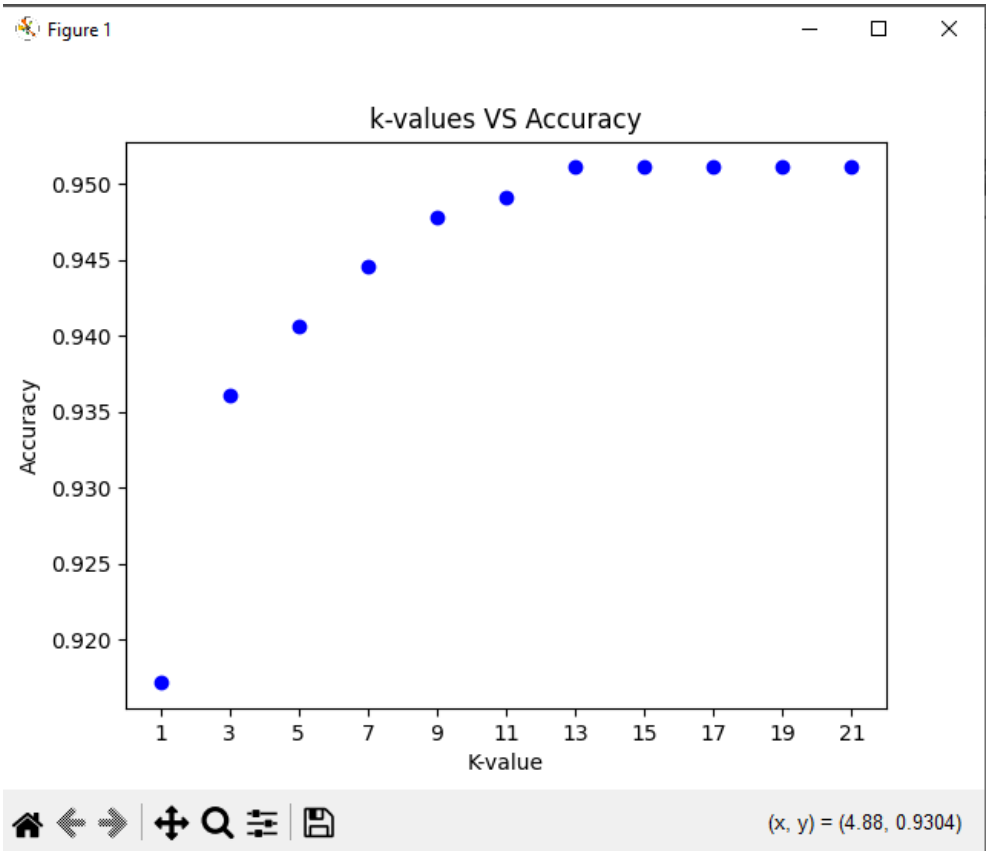

```
K=1  0.9217221135029354
K=3  0.9419439008480104
K=5  0.9458577951728636
K=7  0.949119373776908
K=9  0.9510763209393346
K=11  0.9510763209393346
K=13  0.9510763209393346
K=15  0.9510763209393346
K=17  0.9510763209393346
K=19  0.9510763209393346
K=21  0.9510763209393346
Best K-value is  9
Accuracy with PCA2: 0.9510763209393346
```

With 3 dimensions.



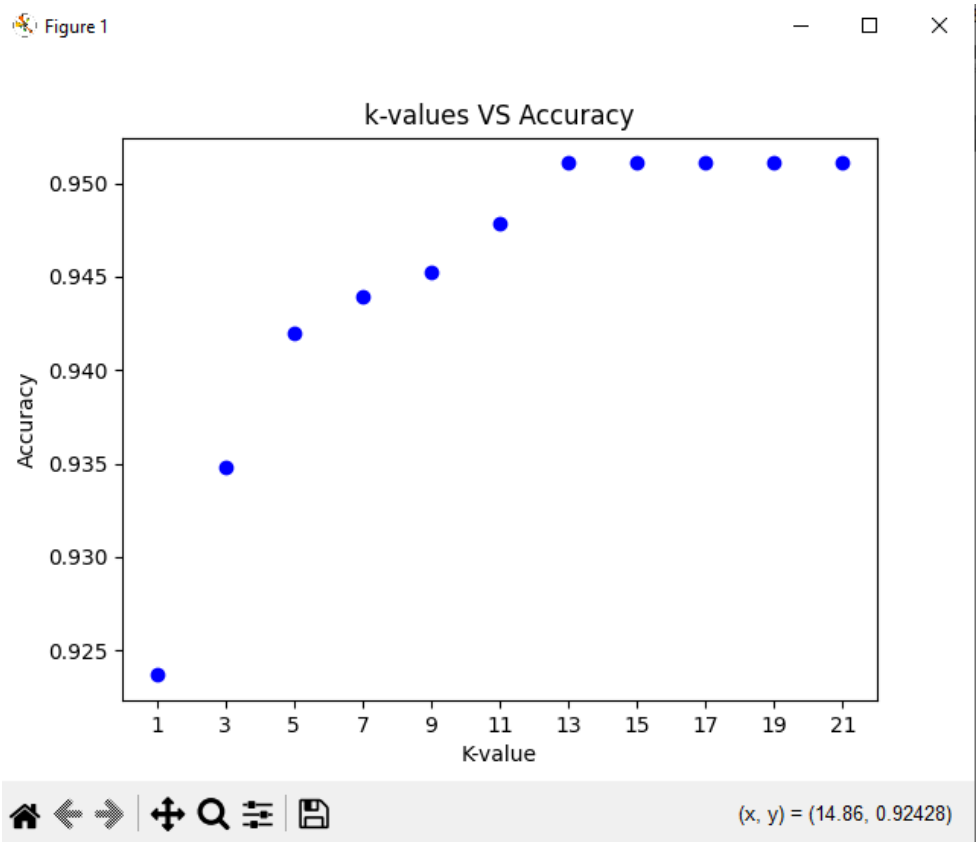

```
K=1  0.9158512720156555
K=3  0.9328114807566862
K=5  0.9406392694063926
K=7  0.9465101108936725
K=9  0.9465101108936725
K=11  0.9504240052185258
K=13  0.9504240052185258
K=15  0.9510763209393346
K=17  0.9510763209393346
K=19  0.9510763209393346
K=21  0.9510763209393346
Best K-value is  15
Accuracy with PCA3: 0.9510763209393346
```

With 4 dimensions.



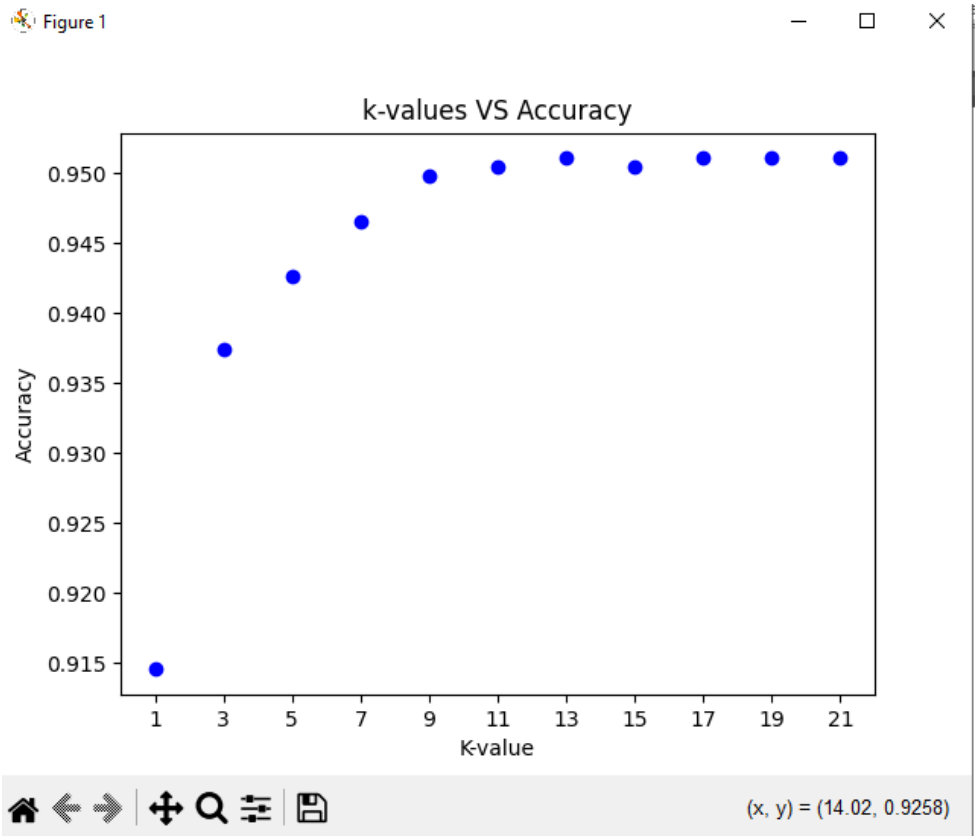

```
K=1  0.9171559034572733
K=3  0.9360730593607306
K=5  0.9406392694063926
K=7  0.9445531637312459
K=9  0.9478147423352903
K=11  0.949119373776908
K=13  0.9510763209393346
K=15  0.9510763209393346
K=17  0.9510763209393346
K=19  0.9510763209393346
K=21  0.9510763209393346
Best K-value is  13
Accuracy with PCA4: 0.9510763209393346
```

With 5 dimensions.



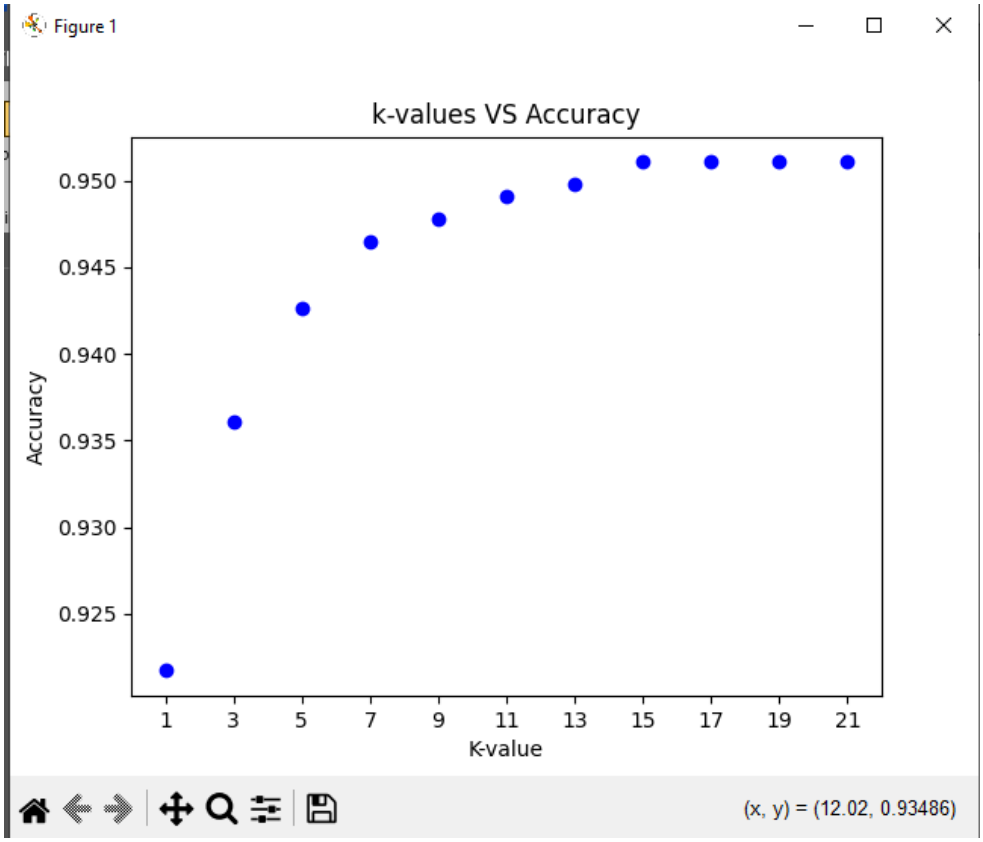

```
K=1  0.923679060665362
K=3  0.9347684279191129
K=5  0.9419439008480104
K=7  0.943900848010437
K=9  0.9452054794520548
K=11  0.9478147423352903
K=13  0.9510763209393346
K=15  0.9510763209393346
K=17  0.9510763209393346
K=19  0.9510763209393346
K=21  0.9510763209393346
Best K-value is  13
Accuracy with PCA5: 0.9510763209393346
```

With 6 dimensions.



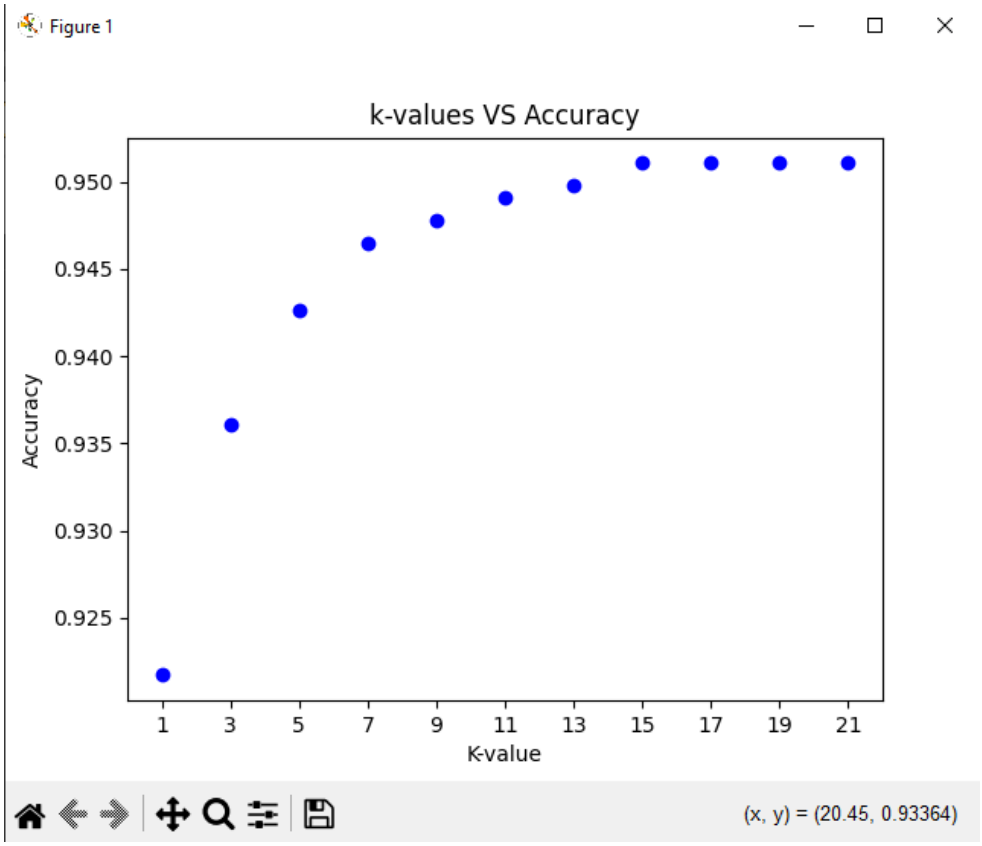

```
K=1  0.9145466405740378
K=3  0.9373776908023483
K=5  0.9425962165688193
K=7  0.9465101108936725
K=9  0.9497716894977168
K=11  0.9504240052185258
K=13  0.9510763209393346
K=15  0.9504240052185258
K=17  0.9510763209393346
K=19  0.9510763209393346
K=21  0.9510763209393346
Best K-value is  13
Accuracy with PCA6: 0.9510763209393346
```

With 7 dimensions.



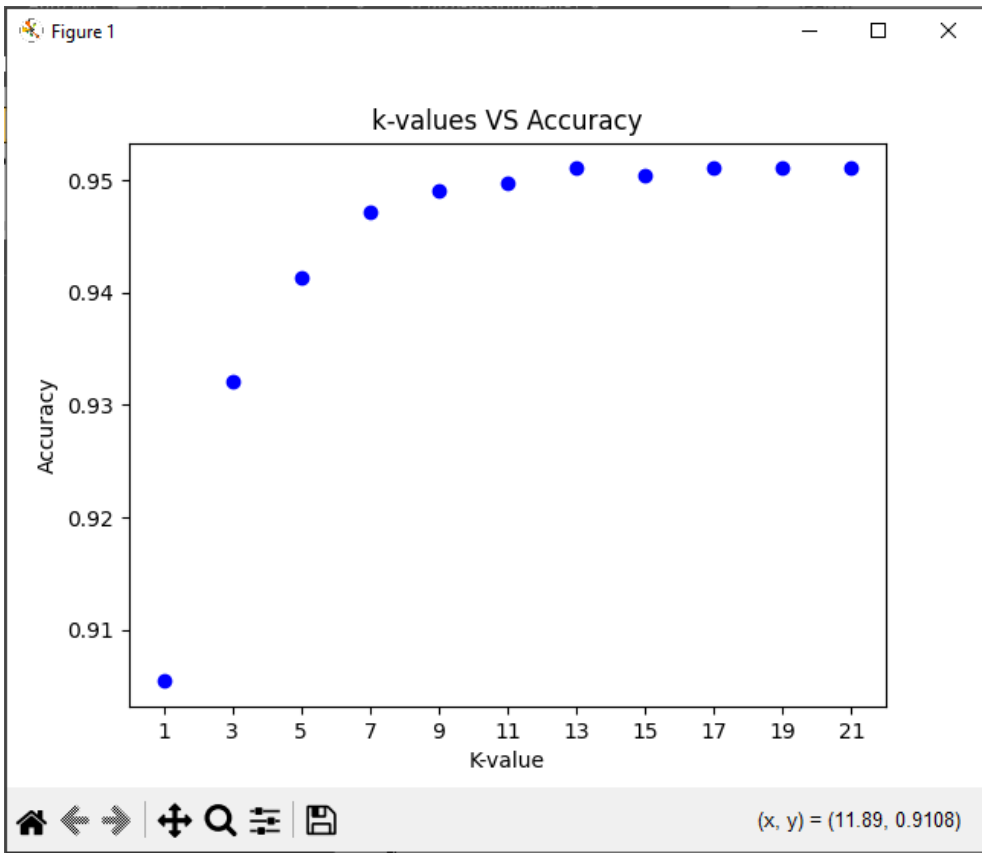

```
K=1  0.918460534898891
K=3  0.9380300065231572
K=5  0.943900848010437
K=7  0.9471624266144814
K=9  0.9484670580560991
K=11  0.9510763209393346
K=13  0.9504240052185258
K=15  0.9510763209393346
K=17  0.9510763209393346
K=19  0.9510763209393346
K=21  0.9510763209393346
Best K-value is  11
Accuracy with PCA7: 0.9510763209393346
```

With 8 dimensions.




```
K=1  0.9217221135029354
K=3  0.9360730593607306
K=5  0.9425962165688193
K=7  0.9465101108936725
K=9  0.9478147423352903
K=11  0.949119373776908
K=13  0.9497716894977168
K=15  0.9510763209393346
K=17  0.9510763209393346
K=19  0.9510763209393346
K=21  0.9510763209393346
Best K-value is  15
Accuracy with PCA8: 0.9510763209393346
```

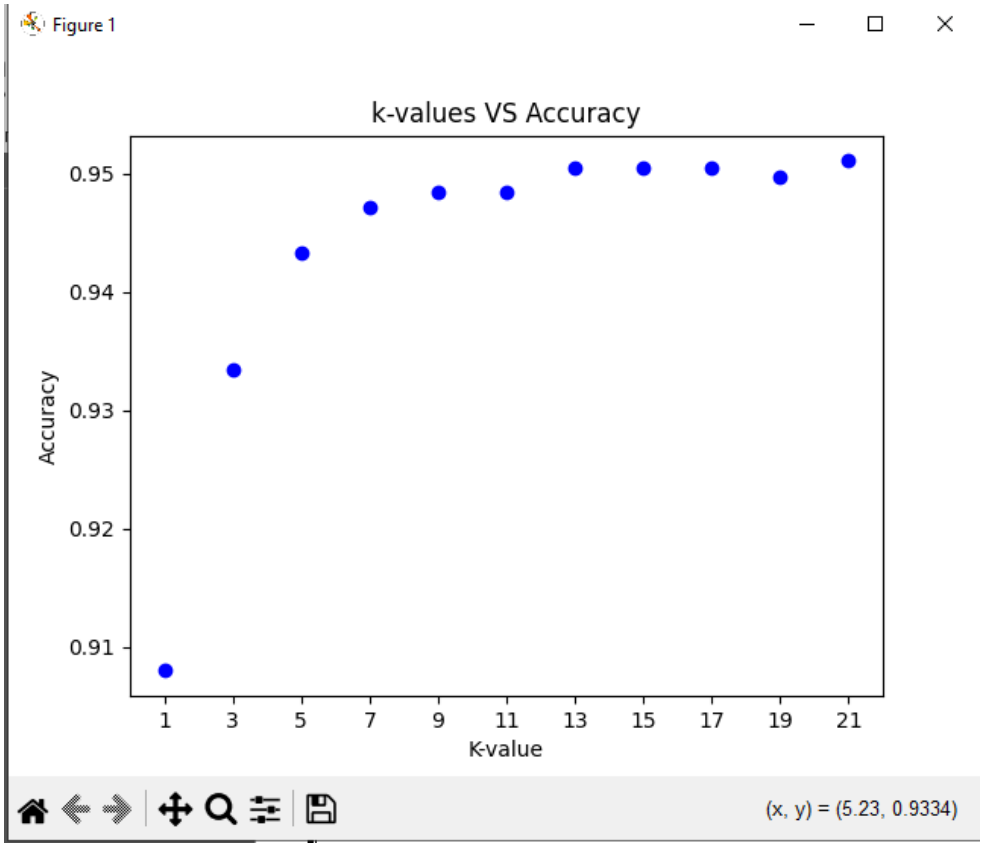
With 9 dimensions.




```
K=1  0.9054142204827136
K=3  0.9321591650358774
K=5  0.9412915851272016
K=7  0.9471624266144814
K=9  0.949119373776908
K=11  0.9497716894977168
K=13  0.9510763209393346
K=15  0.9504240052185258
K=17  0.9510763209393346
K=19  0.9510763209393346
K=21  0.9510763209393346
Best K-value is  13
Accuracy with PCA9: 0.9510763209393346
```

With 10 dimensions.

```
K=1  0.9125896934116112
K=3  0.9360730593607306
K=5  0.9425962165688193
K=7  0.9452054794520548
K=9  0.9471624266144814
K=11  0.9478147423352903
K=13  0.9497716894977168
K=15  0.949119373776908
K=17  0.9504240052185258
K=19  0.9504240052185258
K=21  0.9504240052185258
Best K-value is  17
Accuracy with PCA10: 0.9504240052185258
```



Pca with 1 dimension gives us high accuracy.

```
Selected-PCA Dimensions are  1
```

LDA:

```
LDA:
K=1  0.9080234833659491
K=3  0.9334637964774951
K=5  0.9432485322896281
K=7  0.9471624266144814
K=9  0.9484670580560991
K=11  0.9484670580560991
K=13  0.9504240052185258
K=15  0.9504240052185258
K=17  0.9504240052185258
K=19  0.9497716894977168
K=21  0.9510763209393346
Best K-value is  21
Accuracy with LDA: 0.9510763209393346
```

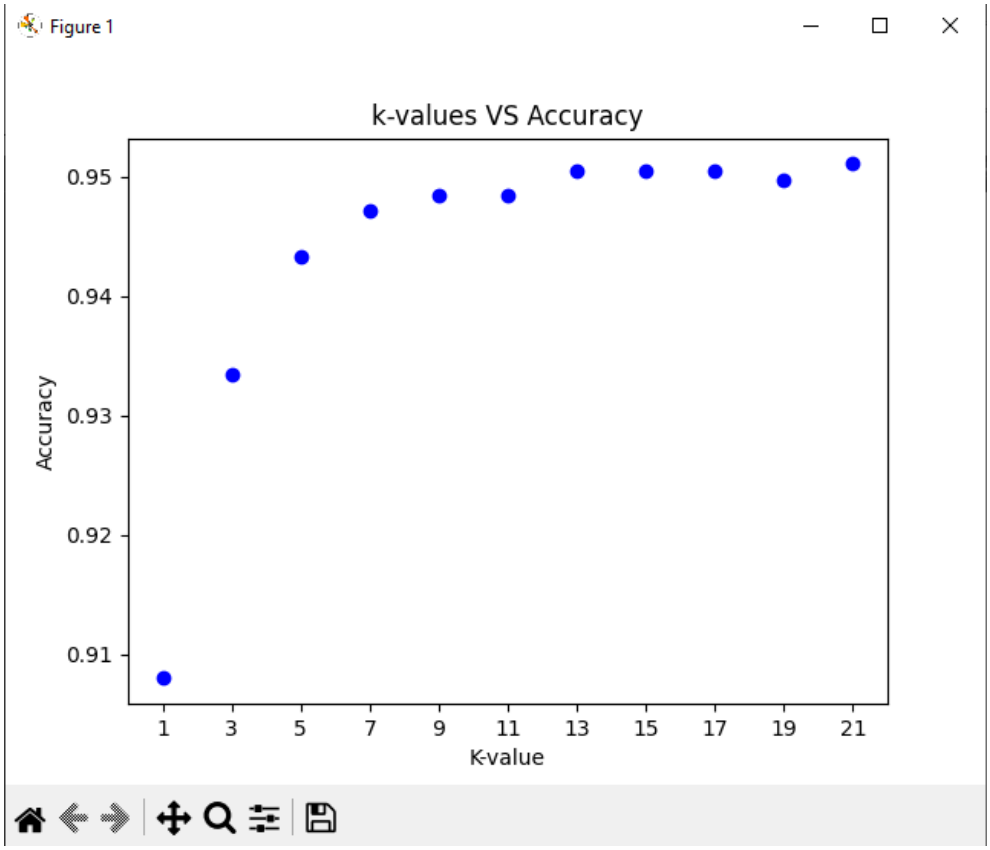


Figure 1
k-values VS Accuracy
Accuracy
0.95
0.94
0.93
0.92
0.91
1
3
5
7
9
11
13
15
17
19
21
K-value

t-SNE:

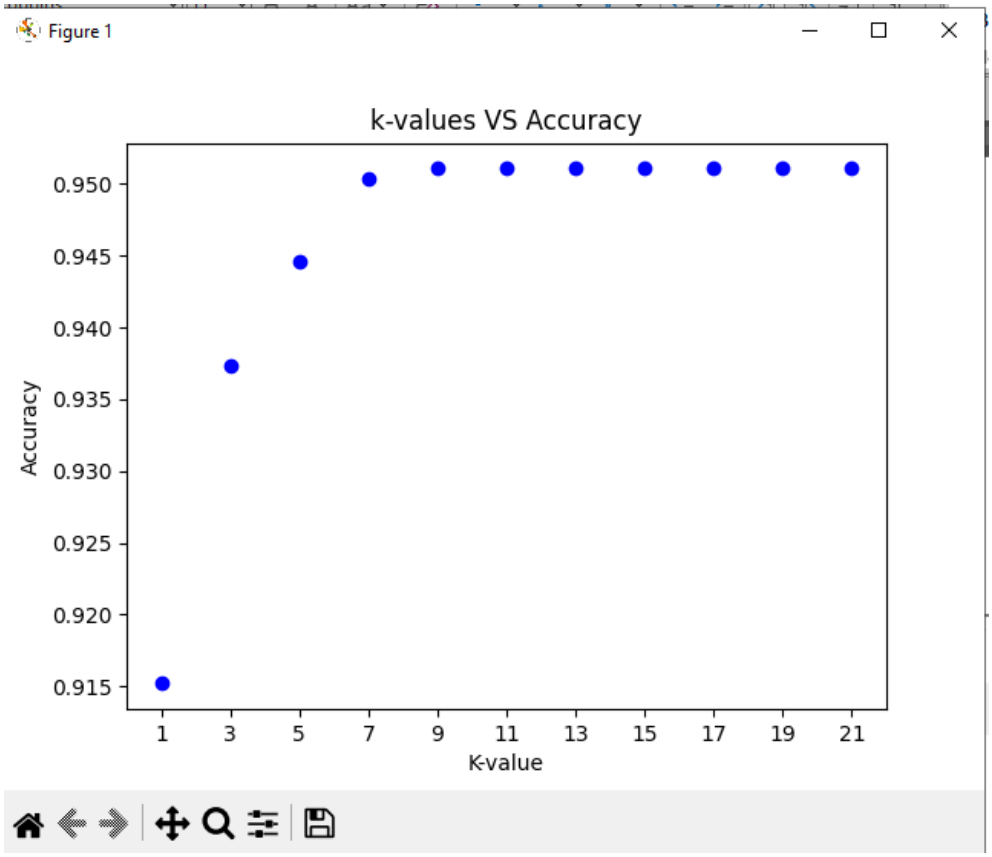


Figure 1
k-values VS Accuracy
Accuracy
0.950
0.945
0.940
0.935
0.930
0.925
0.920
0.915
1
3
5
7
9
11
13
15
17
19
21
K-value

```
t_SNE:
K=1  0.9151989562948467
K=3  0.9373776908023483
K=5  0.9445531637312459
K=7  0.9504240052185258
K=9  0.9510763209393346
K=11  0.9510763209393346
K=13  0.9510763209393346
K=15  0.9510763209393346
K=17  0.9510763209393346
K=19  0.9510763209393346
K=21  0.9510763209393346
Best K-value is  9
Accuracy with tSNE:  0.9510763209393346
```