أكاديمية سدايا
SDAIA Academy

SMART AGRICULTURE

Provide by :
Eng.Nouf Alswuida
T5 Bootcamp

PROBLEM

The goal of this project was to use classification models to predict the operating condition of irrigation field (Irrigation sensors) in smart farm depended on the soil humidity and another features ,in order to help improve solution that will help farmers prepare their irrigation schedules more efficiently and that's will improve automatic irrigate without human intervention . This model will be train on raspberry pi microcontroller with concept of Deep Learning to be the server of the other clients .

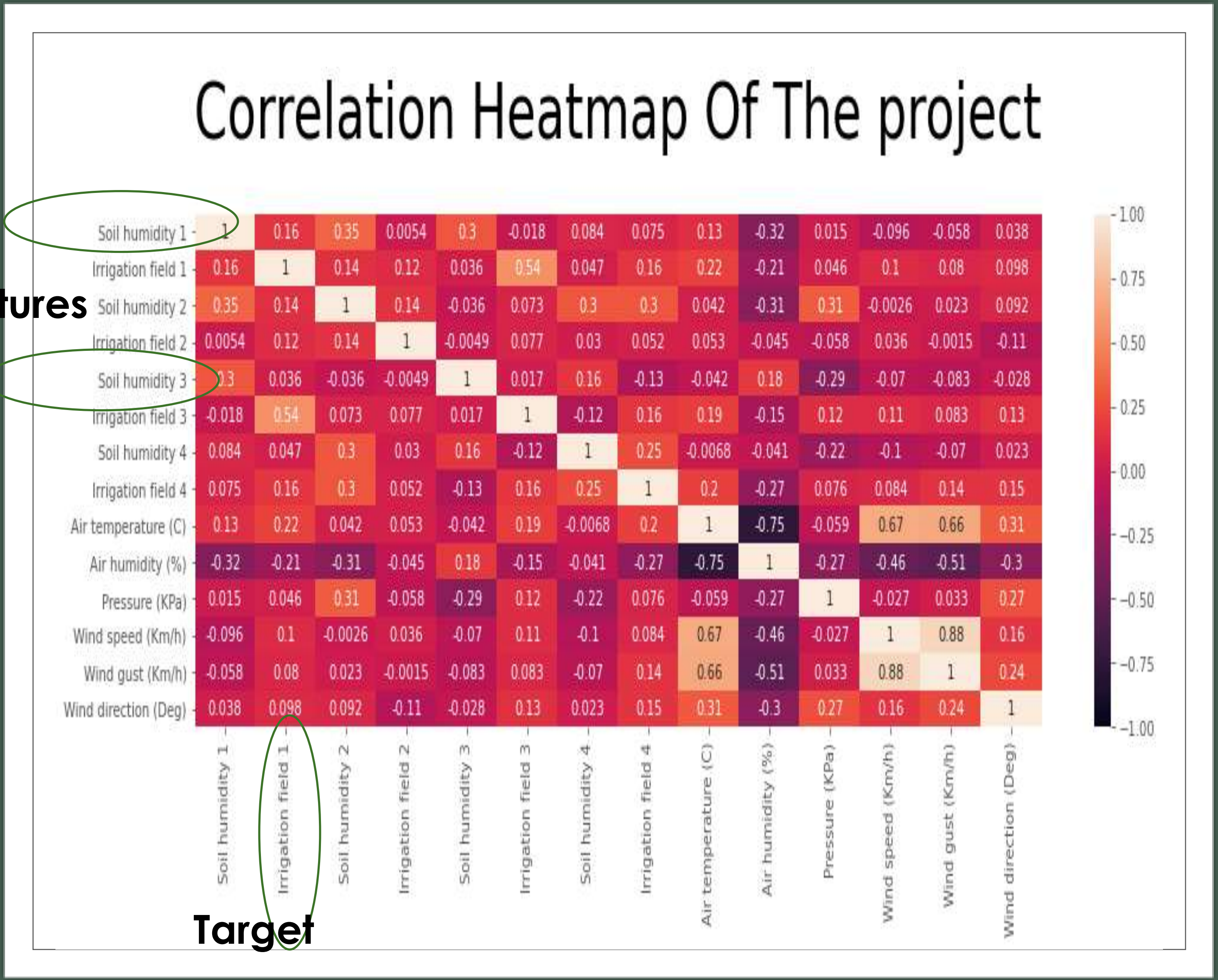
EDA

ANALYZE

INVESTIGATE

INTERESTING RELATIONS

Relationships

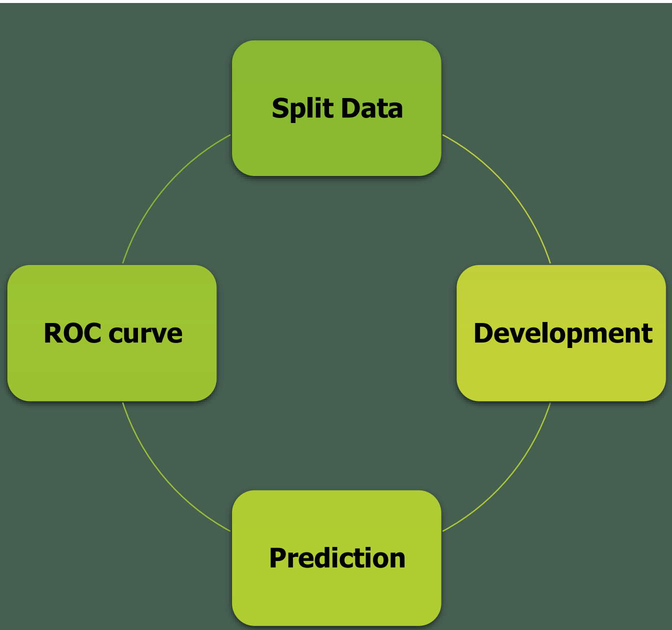


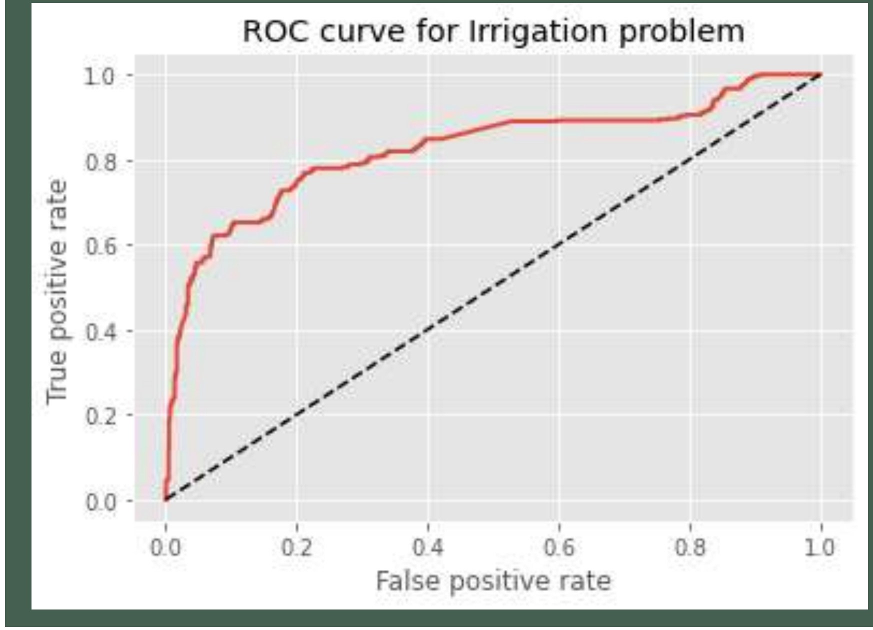
	Soil humidity 1	Irrigation field 1	Soil humidity 2	Irrigation field 2	Soil humidity 3	Irrigation field 3	Soil humidity 4	Irrigation field 4	Air temperature (C)	Air humidity (%)	Pressure (KPa)	Wind speed (Km/h)	Wind gust (Km/h)	Wind direction (Deg)
Soil humidity 1	1	0.16	0.35	0.0054	0.3	-0.018	0.084	0.075	0.13	-0.32	0.015	-0.096	-0.058	0.038
Irrigation field 1	0.16	1	0.14	0.12	0.036	0.54	0.047	0.16	0.22	-0.21	0.046	0.1	0.08	0.098
Soil humidity 2	0.35	0.14	1	0.14	-0.036	0.073	0.3	0.3	0.042	-0.31	0.31	-0.0026	0.023	0.092
Irrigation field 2	0.0054	0.12	0.14	1	-0.0049	0.077	0.03	0.052	0.053	-0.045	-0.058	0.036	-0.0015	-0.11
Soil humidity 3	0.3	0.036	-0.036	-0.0049	1	0.017	0.16	-0.13	-0.042	0.18	-0.29	-0.07	-0.083	-0.028
Irrigation field 3	-0.018	0.54	0.073	0.077	0.017	1	-0.12	0.16	0.19	-0.15	0.12	0.11	0.083	0.13
Soil humidity 4	0.084	0.047	0.3	0.03	0.16	-0.12	1	0.25	-0.0068	-0.041	-0.22	-0.1	-0.07	0.023
Irrigation field 4	0.075	0.16	0.3	0.052	-0.13	0.16	0.25	1	0.2	-0.27	0.076	0.084	0.14	0.15
Air temperature (C)	0.13	0.22	0.042	0.053	-0.042	0.19	-0.0068	0.2	1	-0.75	-0.059	0.67	0.66	0.31
Air humidity (%)	-0.32	-0.21	-0.31	-0.045	0.18	-0.15	-0.041	-0.27	-0.75	1	-0.27	-0.46	-0.51	-0.3
Pressure (KPa)	0.015	0.046	0.31	-0.058	-0.29	0.12	-0.22	0.076	-0.059	-0.27	1	-0.027	0.033	0.27
Wind speed (Km/h)	-0.096	0.1	-0.0026	0.036	-0.07	0.11	-0.1	0.084	0.67	-0.46	-0.027	1	0.88	0.16
Wind gust (Km/h)	-0.058	0.08	0.023	-0.0015	-0.083	0.083	-0.07	0.14	0.66	-0.51	0.033	0.88	1	0.24
Wind direction (Deg)	0.038	0.098	0.092	-0.11	-0.028	0.13	0.023	0.15	0.31	-0.3	0.27	0.16	0.24	1

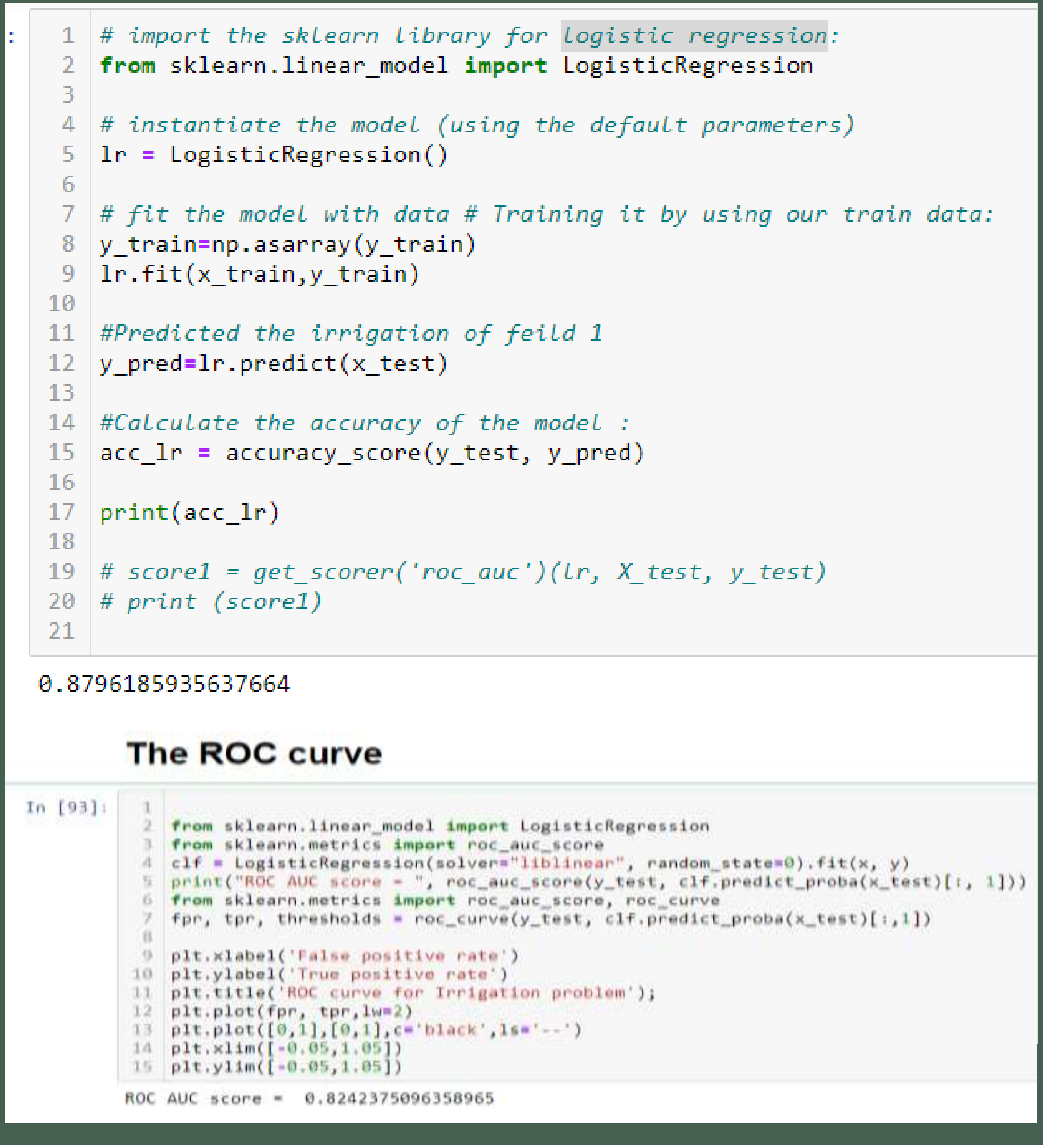
Features

Target

Logistic Regression Model Work & Result :







```
# import the sklearn library for logistic regression:
from sklearn.linear_model import LogisticRegression

# instantiate the model (using the default parameters)
lr = LogisticRegression()

# fit the model with data # Training it by using our train data:
y_train=np.asarray(y_train)
lr.fit(x_train,y_train)

#Predicted the irrigation of feild 1
y_pred=lr.predict(x_test)

#Calculate the accuracy of the model :
acc_lr = accuracy_score(y_test, y_pred)

print(acc_lr)

# score1 = get_scorer('roc_auc')(lr, X_test, y_test)
# print (score1)

```

```
0.8796185935637664
```

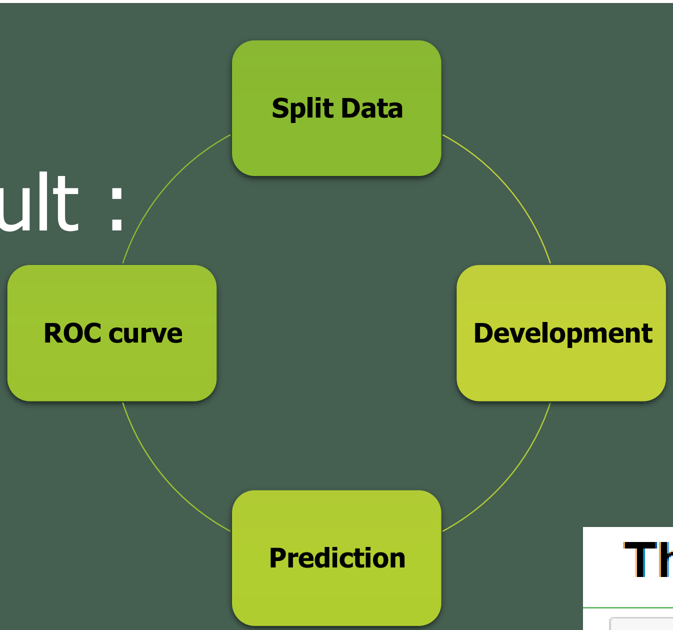
The ROC curve

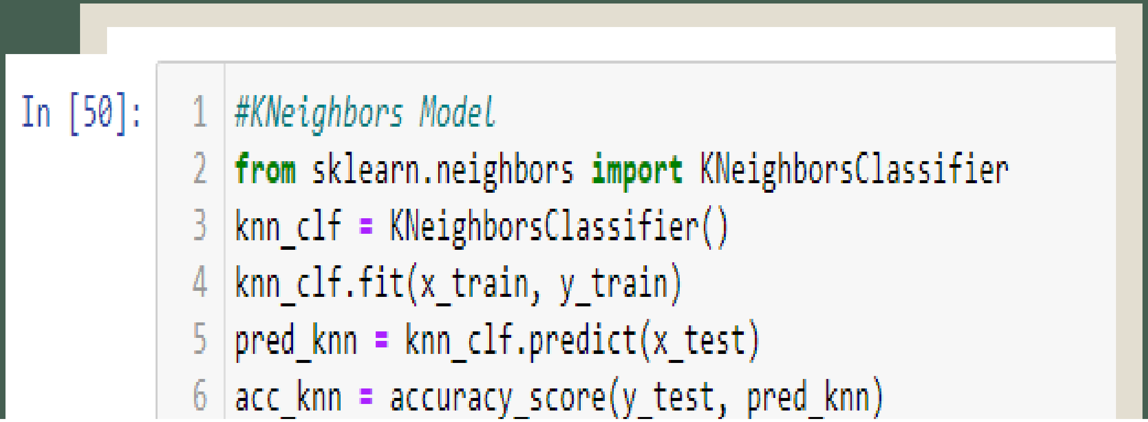
In [93]:

```

from sklearn.linear_model import LogisticRegression
from sklearn.metrics import roc_auc_score
clf = LogisticRegression(solver="liblinear", random_state=0).fit(x, y)
print("ROC AUC score = ", roc_auc_score(y_test, clf.predict_proba(x_test)[:, 1]))
from sklearn.metrics import roc_auc_score, roc_curve
fpr, tpr, thresholds = roc_curve(y_test, clf.predict_proba(x_test)[:,1])

plt.xlabel('False positive rate')
plt.ylabel('True positive rate')
plt.title('ROC curve for Irrigation problem');
plt.plot(fpr, tpr,lw=2)
plt.plot([0,1],[0,1],c='black',ls='--')
plt.xlim([-0.05,1.05])
plt.ylim([-0.05,1.05])
```

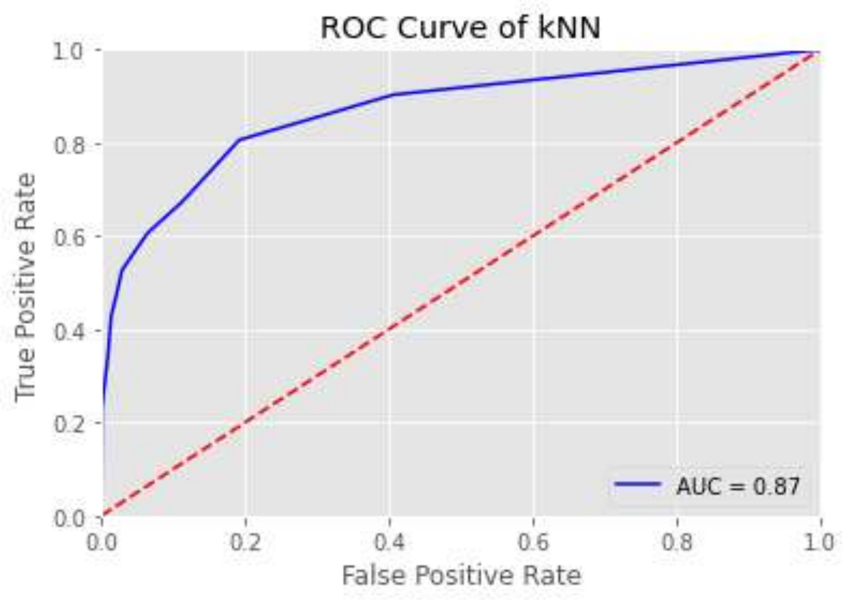
```
ROC AUC score =  0.8242375096358965
```

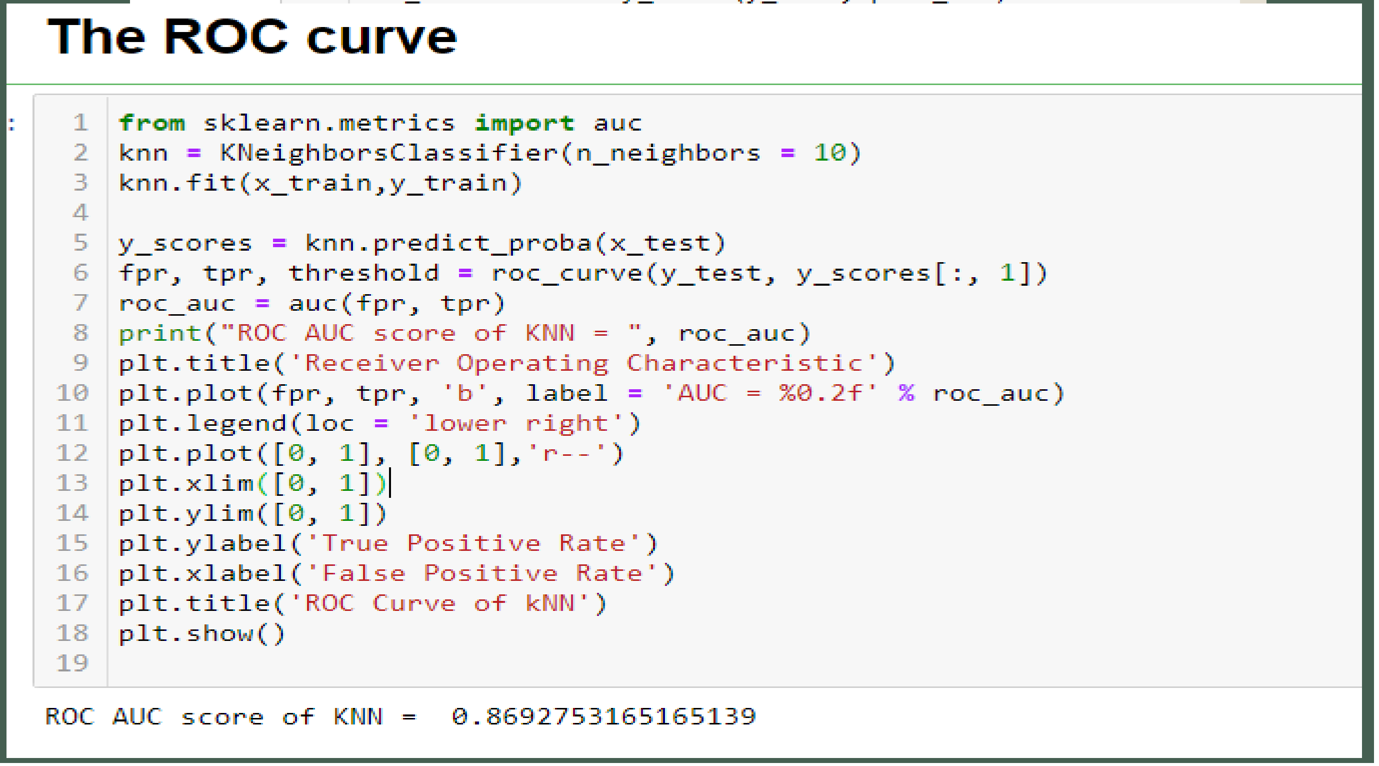
KNN Model Work & Result :





```
In [50]:  #KNeighbors Model
          from sklearn.neighbors import KNeighborsClassifier
          knn_clf = KNeighborsClassifier()
          knn_clf.fit(x_train, y_train)
          pred_knn = knn_clf.predict(x_test)
          acc_knn = accuracy_score(y_test, pred_knn)
```



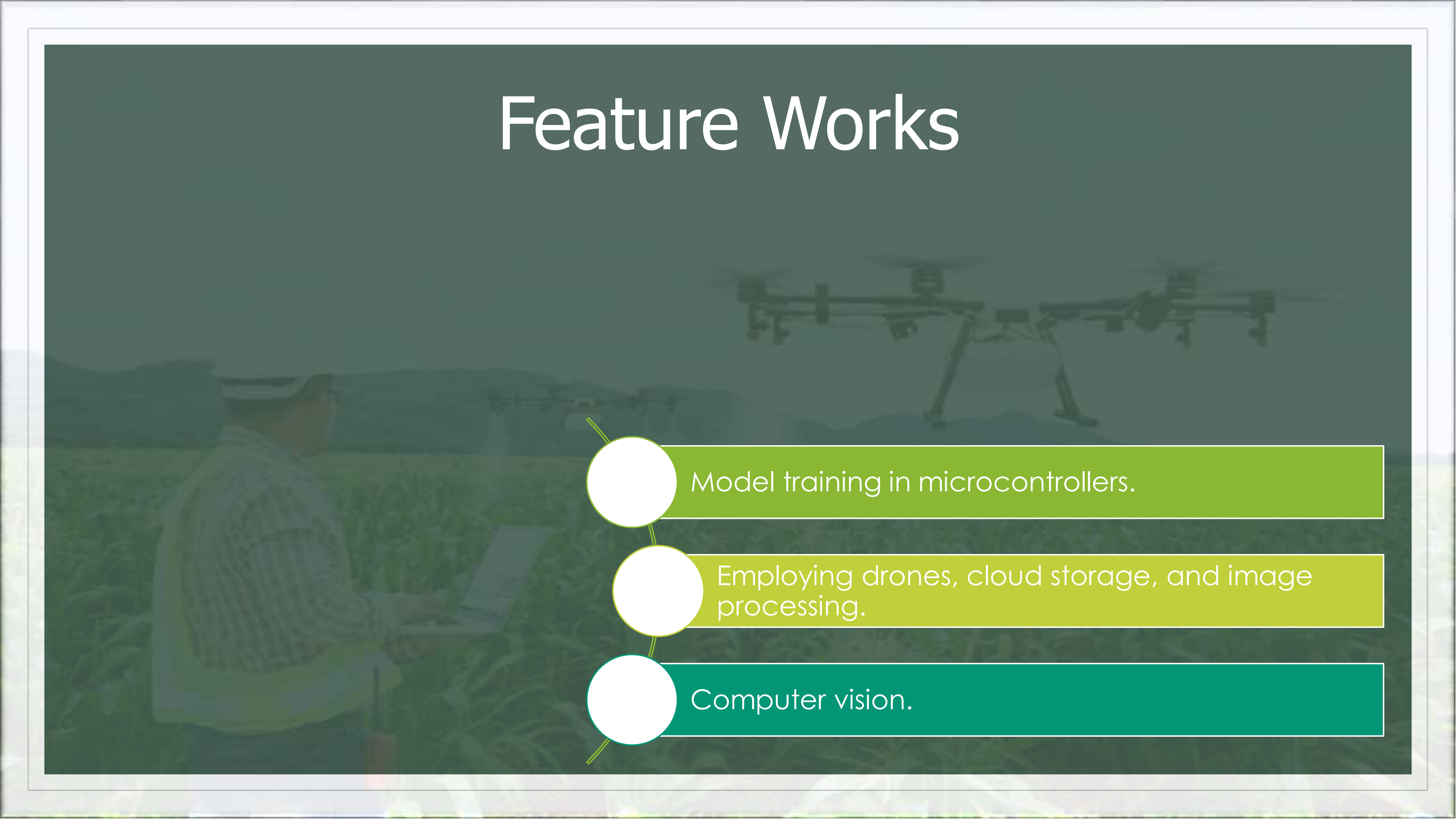


The ROC curve

```
from sklearn.metrics import auc
knn = KNeighborsClassifier(n_neighbors = 10)
knn.fit(x_train,y_train)

y_scores = knn.predict_proba(x_test)
fpr, tpr, threshold = roc_curve(y_test, y_scores[:, 1])
roc_auc = auc(fpr, tpr)
print("ROC AUC score of KNN = ", roc_auc)
plt.title('Receiver Operating Characteristic')
plt.plot(fpr, tpr, 'b', label = 'AUC = %0.2f' % roc_auc)
plt.legend(loc = 'lower right')
plt.plot([0, 1], [0, 1],'r--')
plt.xlim([0, 1])
plt.ylim([0, 1])
plt.ylabel('True Positive Rate')
plt.xlabel('False Positive Rate')
plt.title('ROC Curve of kNN')
plt.show()

```

```
ROC AUC score of KNN =  0.8692753165165139
```

Feature Works



- Model training in microcontrollers.
- Employing drones, cloud storage, and image processing.
- Computer vision.

THANK YOU FOR LISTENING