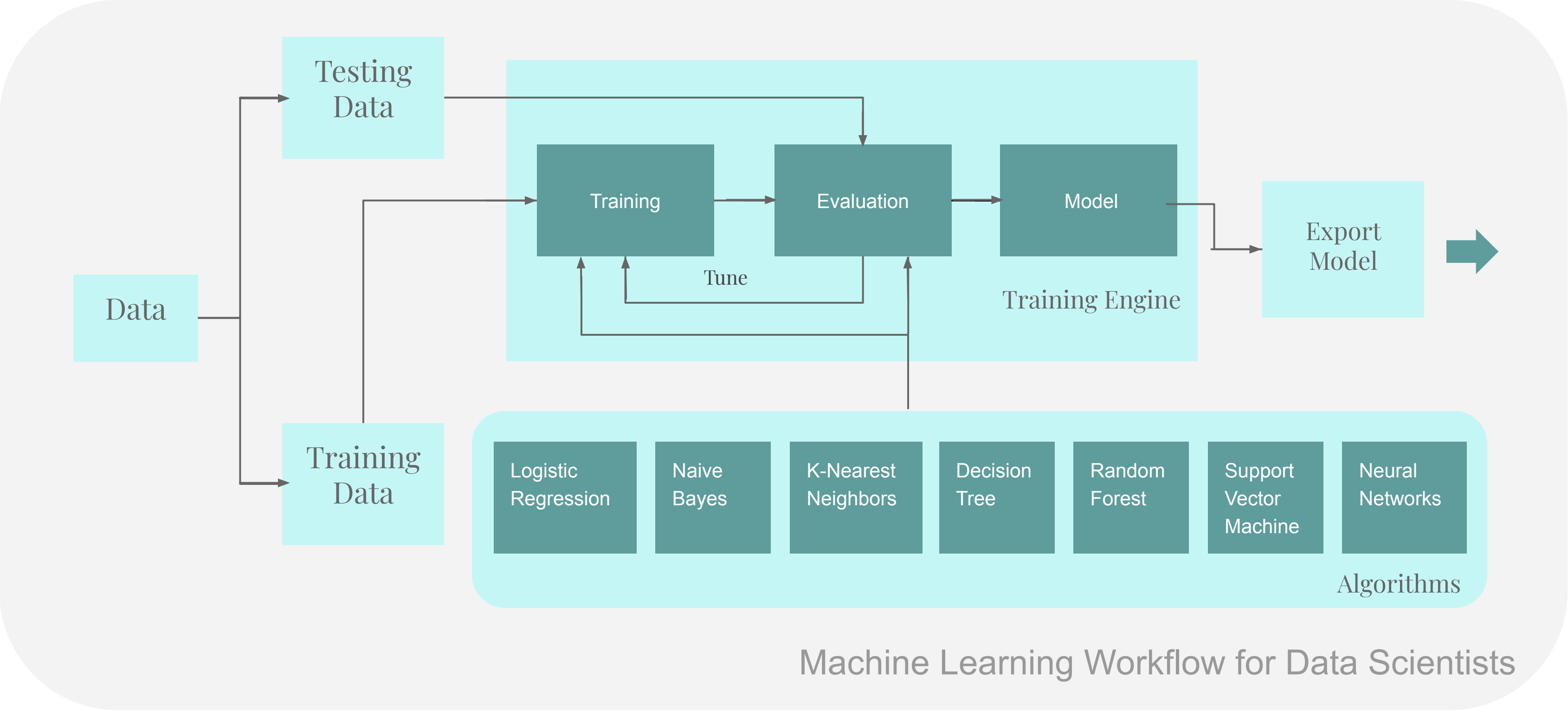
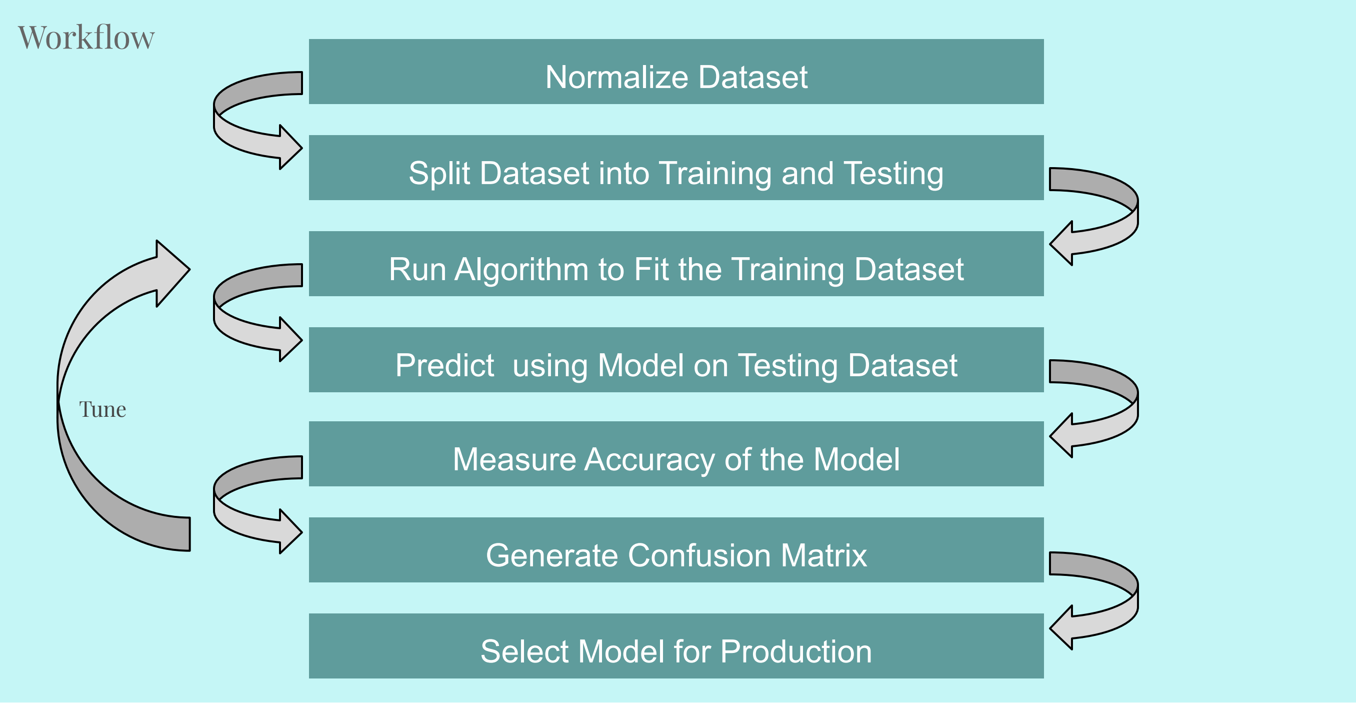
Model Training Evaluation Analysis



Model Training Evaluation Analysis



Workflow

Normalize Dataset

Split Dataset into Training and Testing

Run Algorithm to Fit the Training Dataset

Predict using Model on Testing Dataset

Measure Accuracy of the Model

Generate Confusion Matrix

Select Model for Production

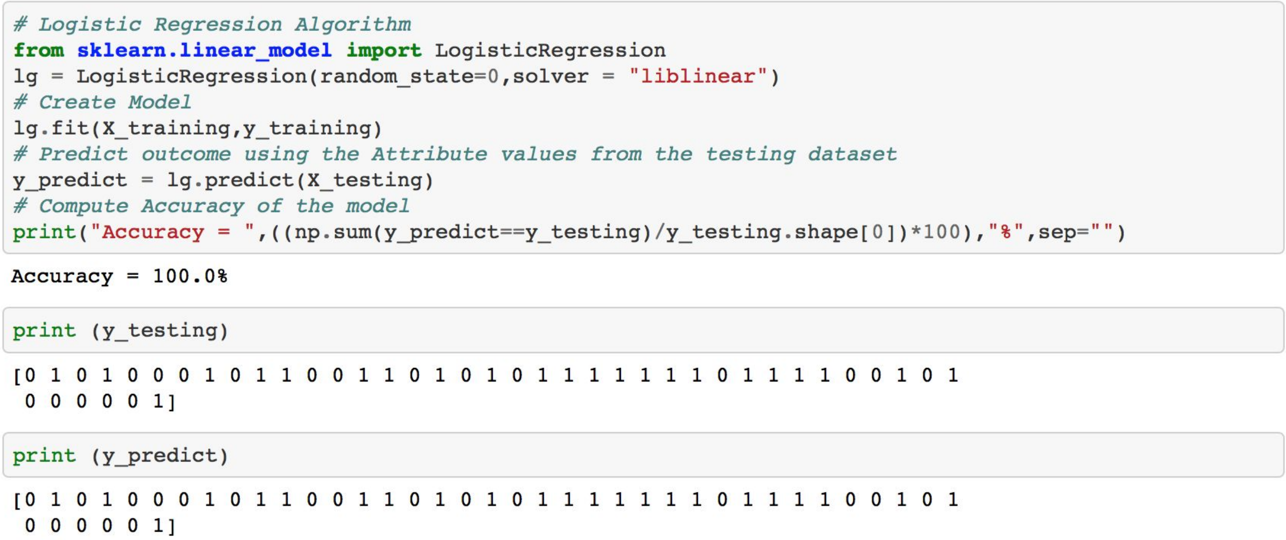
Tune

Model Training Evaluation Analysis

Machine Learning Algorithms

Logistic Regression Model selected for final Production Inferencing.

- *Logistic Regression Algorithm (***sklearn.linear_model.**LogisticRegression)



```
# Logistic Regression Algorithm
from sklearn.linear_model import LogisticRegression
lg = LogisticRegression(random_state=0,solver = "liblinear")
# Create Model
lg.fit(X_training,y_training)
# Predict outcome using the Attribute values from the testing dataset
y_predict = lg.predict(X_testing)
# Compute Accuracy of the model
print("Accuracy = ",((np.sum(y_predict==y_testing)/y_testing.shape[0])*100),"%",sep="")
```

```
Accuracy = 100.0%
```

```
print (y_testing)
```

```
[0 1 0 1 0 0 0 1 0 1 1 0 0 1 1 0 1 0 1 0 1 1 1 1 1 1 1 0 1 1 1 1 0 0 1 0 1
 0 0 0 0 0 1]
```

```
print (y_predict)
```

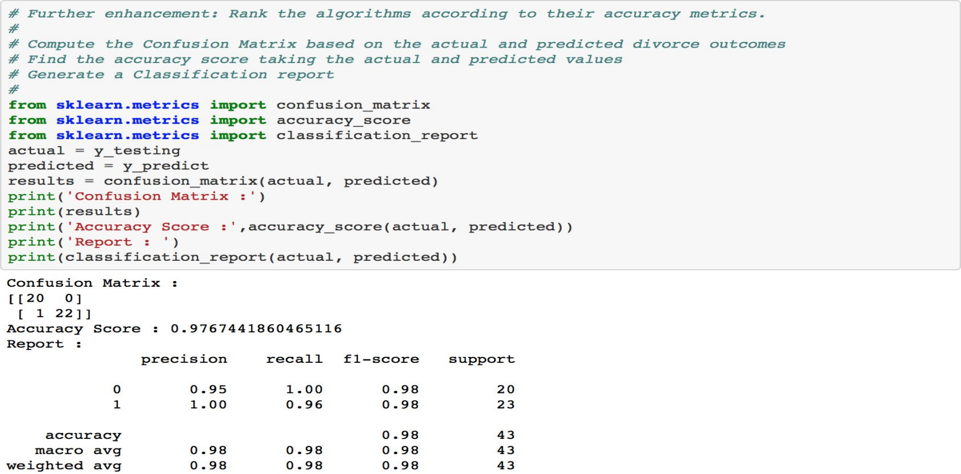
```
[0 1 0 1 0 0 0 1 0 1 1 0 0 1 1 0 1 0 1 0 1 1 1 1 1 1 1 0 1 1 1 1 0 0 1 0 1
 0 0 0 0 0 1]
```

Model Training Evaluation Analysis

Machine Learning Algorithms

- *Naive Bayes Algorithm (***sklearn.naive_bayes.***GaussianNB)*
- *K-Nearest Neighbors (KNN) Algorithm: (***sklearn.neighbors.***KNeighborsClassifier)*
- *Decision Tree Algorithm (***sklearn.tree.***DecisionTreeClassifier)*
- *Random Forest Algorithm (***sklearn.ensemble.***RandomForestClassifier)*
- *Support Vector Machine (SVM) Algorithm (***sklearn.svm***.SVC)*
- *Neural Networks Algorithm (***keras.models***.Sequential)*

Sample Confusion Matrix



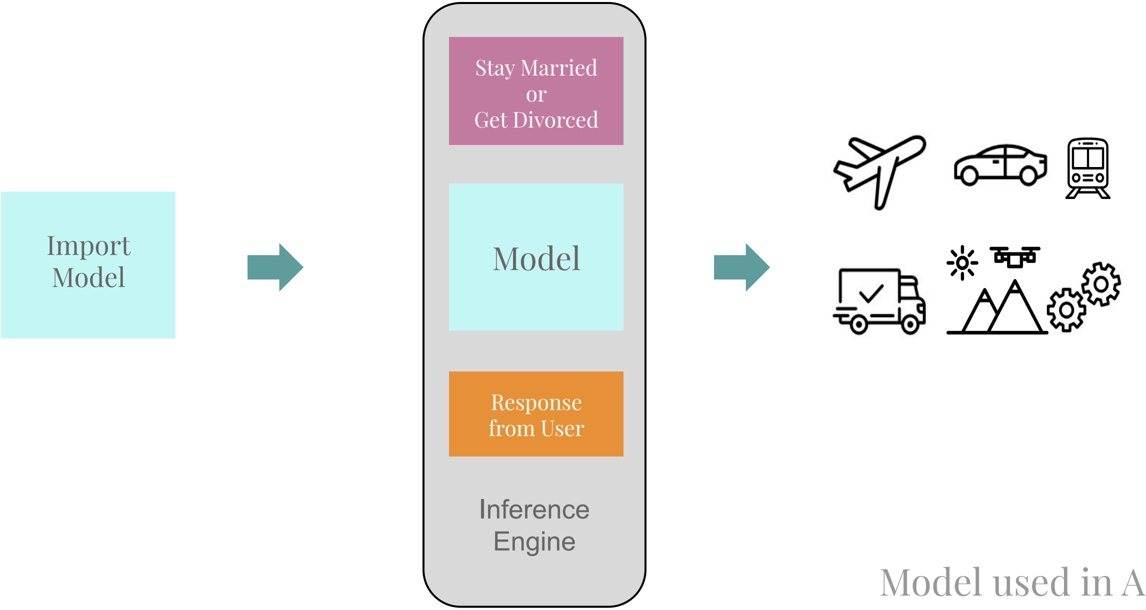
```
# Further enhancement: Rank the algorithms according to their accuracy metrics.
#
# Compute the Confusion Matrix based on the actual and predicted divorce outcomes
# Find the accuracy score taking the actual and predicted values
# Generate a Classification report
#
from sklearn.metrics import confusion_matrix
from sklearn.metrics import accuracy_score
from sklearn.metrics import classification_report
actual = y_testing
predicted = y_predict
results = confusion_matrix(actual, predicted)
print('Confusion Matrix :')
print(results)
print('Accuracy Score :',accuracy_score(actual, predicted))
print('Report : ')
print(classification_report(actual, predicted))
```

```
Confusion Matrix :
[[20  0]
 [ 1 22]]
Accuracy Score : 0.9767441860465116
Report :
              precision    recall  f1-score   support

           0       0.95      1.00      0.98        20
           1       1.00      0.96      0.98        23

    accuracy                           0.98        43
   macro avg       0.98      0.98      0.98        43
weighted avg       0.98      0.98      0.98        43
```

Inferencing in the Field



Import Model

Stay Married or Get Divorced

Model

Response from User

Inference Engine

Model used in Applications