ML Model Selection

Predicting Bank Lead Conversion

Authored by: Keshava G

Model performance dashboard

Summary of performance metrics for trained ML models

Four different machine learning models were trained, and their performance was evaluated based on parameters such as prediction accuracy, time taken for training and prediction, space utilization by the model and whether the model is available to give us the most important features affecting the lead conversion or not.
Below are the results obtained.

	Test-set accuracy	Training time	Prediction time	Model storage space	Feature imp (Y/N)
Logistic regression	0.8325	1.7787871360778809	0.003384828567504883	1425	Y
Decision tree	0.953	0.04671525955200195	0.003238677978515625	16081	Y
XGBoost	0.972	0.8535799980163574	0.013007640838623047	139092	Y
SVM model	0.8505	0.8583385944366455	0.3113975524902344	294317	N

Models fulfilling the given requirements:

1. Test set prediction accuracy should be more than 95% - Decision tree and XGBoost
2. The model should clearly tell the most important features. – Logistic regression, decision tree and XGBoost.
3. Prediction time should be less than 0.1 – Logistic regression, decision tree and XGBoost.
4. If all three conditions specified above are met, the model with lowest storage size should be selected. – Logistic regression

Recommendation - Logistic Regression

Python code for training and evaluating Logistic Regression model

```
import time
from sklearn.linear_model import LogisticRegression
logreg = LogisticRegression(max_iter=1000)

start_time = time.time()
logreg.fit(X_train_smote, y_train_smote)
logreg_train_time = time.time() - start_time
logreg_train_time

from sklearn.metrics import accuracy_score
start_time = time.time()
y_pred_logreg = logreg.predict(X_test)
logreg_predict_time = time.time() - start_time
logreg_accuracy = accuracy_score(y_test, y_pred_logreg)
logreg_accuracy, logreg_predict_time
```

Serialize Logistic Regression Model

```
1. logreg_stream = pickle.dumps(logreg)
2. logreg_size = sys.getsizeof(logreg_stream)
logreg_coef = abs(logreg.coef_[0])

top_2_logreg_features = X.columns[logreg_coef.argsort()[-2:][::-1]]
top_2_logreg_features
```