



Fraud Watch

DEC 2, 2019

XIAOHUA SHI | ZHITONG 'MIA' XIE | AJINKYA SHETH

ABSTRACT

Unauthorised or Fraudulent Transactions plague the banking and finance sector. In this report, we choose to explore how machine learning can be used to detect fraud in bank transactions. Since real-world financial data is highly protected and secured, we use a synthetic dataset borrowed from Kaggle to make our case. Our machine learning solution is powered by Microsoft Azure's ML Studio. ML Studio is a machine learning product which simplifies the task of developing ML models using a GUI. In this report, we stress on how ML Studio can automate the task of deploying highly scalable data solutions to help the banking/finance business. In order to make our case relatable, we institute a fictional company 'Freya Group' and assume it to be a top financial services provider in the country. Our project 'Fraud Watch' is a machine learning solution about fraud detection for Freya Group.

CONTENTS

Introduction	5
Strategy	5
Why the company wants to transform?	5
Business Problem	6
Impact on the Existing Business Model	6
Phases of Transformation	7
Scope	7
Teams	8
Design	8
Methods	10
Web App	15
Results	16
Discussion	17
Learnings	18
Assumptions	18
Future Work	18
Related Work or Research	19

Introduction

Our team chose Banking/Financial industry as topic. Banking industry has been adopting digital transformation strategy to meet the ever increasing needs and expectations of customers. Due to ease of access and convenience of using web; online services and mobile apps are being utilized more as compared to the traditional brick-and-mortar banking portals such as ATM and branches. However, as more and more transactions are completed through devices and networks, fraudulent transactions become a serious and inevitable problem that threatens the health of the banking industry. Our digital transformation solution aims to provide banks' customers with highly reliable fraud detection system and reduce the risk of losses caused by unauthorized transactions.

Strategy

Freya Group is a top US-based financial services provider which has been growing tremendously since last few years. One of their key services is mobile banking. Freya Group has been using on-premise servers to handle most of their mobile banking transactions. Users are allowed to use mobile or web application to complete transactions and payments, change personal information such as address, and view activity history in order to manage income as well as expenditure. They also have a regression-based fraud detection system to detect fraudulent transactions based on variables like transaction amount, transaction type, transaction original account and destination account. However, the algorithm to detect fraud is inefficient and ineffective. The group executives are realizing the limitations of the company's current digital system. The group has already established its presence in 10 countries and is planning to expand in more. In order to support the growing digital requirements, the group executives want to bring a digital transformation in the company by availing power of cutting edge technologies developed by Microsoft.

Why the company wants to transform?

As mentioned before, with the expansion of company size and the growing customers' needs, the company needs to improve their current digital transformation to mitigate the risk of fraud transactions, improve efficiency and better serve customers. Adopting a cloud service such as TIBCO Cloud Live Apps –

an application development platform, can help the business to build a customized solution to detect fraud and meet increased regulations.

Business Problem

Based on our exploratory data analysis on the current fraud detection data, the rate of successful fraud detection does not meet the company's expectations. In our dataset, 0.13% of total transactions in the data are fraud transactions and only 0.195% of fraud transactions are successfully identified which means that the system does not work well right now. Thus, a new approach that uses technology like machine learning and advanced analytics is needed to augment the control of fraud detection. The group needs to aggressively leverage the potential provided by Digital, Mobile and the Cloud. They should take measures like building a **robust analytics layer** on top of aggregated customer and industry data to aid in data-driven measures (Bharathi, 2019).

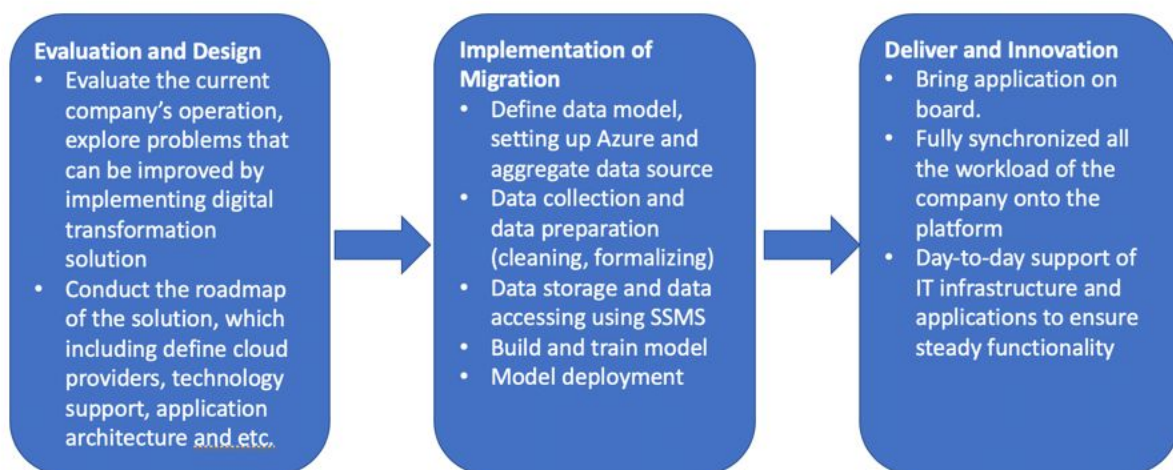
Impact on the Existing Business Model

The group needs a highly elastic infrastructure and thus is thinking of going for public cloud. The other reasons for which are security and disaster recovery. The executives have decided on the extent of infrastructure, the user experience and ease of deployment as top priorities in choosing a cloud vendor. Microsoft's Azure has been selected thus. As a part of initial rollout plan, the company wants to offload their fraud detection system on cloud. In doing so, they plan to scrap their legacy fraud detection system and instead modernize using Azure's cutting-edge technology stack. They intend to use in-built Azure's data storage, machine learning and reporting capabilities for the same. The new system will exploit Azure's high processing capabilities to detect fraud quickly and more accurately thereby giving a better service to their customers.

The company's earlier approach to business has always been infrastructure first. The company would expand, provide new services or market themselves to new customers only if there is a reliable physical IT infrastructure in place. Since setting up IT infrastructure is extremely expensive, the company had to consider each and every business risk. However, with cloud adoption, the company has more flexibility and can adapt itself to changing market conditions quickly by reducing the reliance on the hardware. This solution changes their business model enabling them to venture out to new markets quickly with minimal risks with respect to IT infrastructure.

After fully adopting the aforementioned digital transformation solution. The company aims to be more creative in expanding the boundaries of their existing business model by utilizing digital technology and big data analytics in the future. Traditional bank approach has been about introducing banking products. For example, a customer with a checking account would be encouraged to consider a personal line of credit, a home-improvement loan, or a bank credit card (Khanna & Martins, 2018). Modern approach includes exploring beyond the traditional core services by capturing customer data and identifying patterns giving rise to new business opportunities such as developing money management application, utilizing data to explore B2B business partners, etc.

Phases of Transformation



Scope

Our dataset is derived from Kaggle and it is a synthetic financial dataset generated by PaySim for fraud detection. It is a csv file with 1 header row. PaySim simulates mobile money transactions based on a sample of real transactions extracted from

one month of financial logs from a mobile money service implemented in US. In our case, the synthetic dataset is scaled down to 1/4 of the original dataset or about 250k rows.

Teams

Tech Team plays the most significant role because all digital transformations involve technology. With the focus of improving current fraud detection accuracy, the tech team needs someone who is proficient in machine learning and big data analytics. The tech team also needs engineers who have experience in cloud development so they can build in-house solution or choose the right cloud service provider. In addition to the tech team, management team is also important because of their role of control and process management. The management team is responsible for drawing digital transformation roadmap including ownership, investments, KPI and etc., and make sure the digital transformation solution is aligned with business goals.

Design

We obtain our data from Kaggle. The data schema is shown as a list below.

- Step: hashed time stamp of each transaction
- Type: the type of transactions, having values [CASH-IN, CASH-OUT, DEBIT, PAYMENT, TRANSFER]
- Amount: amount of the transaction in local currency.
- nameOrig: customer who started the transaction
- nameDest: customer who is the recipient of the transaction
- oldbalanceOrig/oldbalanceDest: initial balance before the transaction for Orig/Dest
- newbalanceOrig/newbalanceDest - initial balance before the transaction for Orig/Dest
- isFraud: 1 for a true fraud transaction and 0 for not.
- isFlaggedFraud: the indicator for the current model performance. 1 for a transaction detected by the current system as a fraud and 0 for not.

After discussion, we determined to apply Azure in our strategy pipeline, which is shown in Figure 1 below.

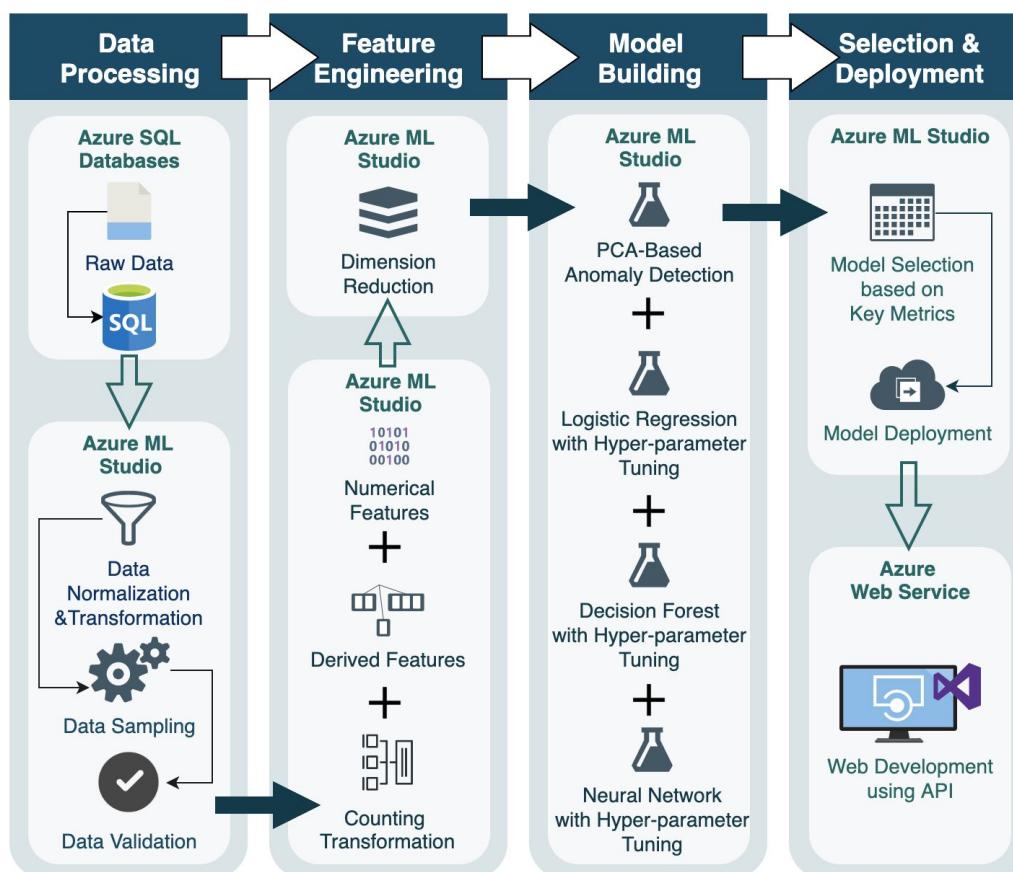


Fig 1. Azure solution pipeline for digital transformation strategy.

As for our evaluation metrics, we evaluated the performance of each model not only based on the traditional data science algorithm metrics such as accuracy, precision, recall, F-score, AUC, average log loss, and training log loss, but also on some specific metrics related to financial fraud detection listed below (AzureML Team, 2015).

- ADR – Fraud Account Detection Rate. The percentage of detected fraud accounts in all fraud accounts.
- VDR - Value Detection Rate. The percentage of monetary savings, assuming the current fraud transaction triggered a blocking action on subsequent transactions, over all fraud losses.
- AFPR - Account False Positive Ratio. The ratio of detected false positive accounts over detected fraud accounts.

Methods

Azure is the key component to our digital transformation solution pipeline due to the fact that it is quite powerful in solving digital transformation issues for industrial world. Azure is included with so many convenient services such as Azure SQL Storage, Azure Machine Learning studio, and Azure Cognitive Services, which makes it an undoubtable choice.

Initially, we acquire our data from Kaggle and then store it in Azure SQL Database for easy and seamless access. Then we proceed with data science modeling using Azure Machine Learning studio, which is a convenient tool for data science model deployment. Our first step is to conduct data processing including data ingestion, data transformation, data normalization, data sampling and data validation. Figure 2 below shows the Azure experiment we conducted for this step. After ingesting the raw data from Azure SQL Server to Azure ML Studio as a csv file, we transform it by removing duplicated rows and filtering the column of 'amount' as less than or equal to 10,000,000. This decision is made based on the results from our previous EDA (Exploratory Data Analysis) that the data is quite sparse when the transaction amount larger than 10,000,000. In order to remove potential outliers, we removed all the data point with 'amount' being larger than 10,000,000. Then we normalized the data by converting the columns of 'isFraud' to integer, cleaning missing data and removing the column of 'isFlaggedFraud', which is not useful in model training and deployment. However, the data is still unbalanced after these operations (the true fraud rate is only 0.109%) and we decided to conduct data sampling. By studying the results from Yuzhou Song's experiments (Song, 2015), We applied the method of combining down-sampling with over-sampling for this case. We first down sampled the non-fraud transactions with down sampling rate 0.08 and then over sampled the fraud transactions using SMOTE with over sampling rate being 300%. The fraud rate in the data became 6.07% after sampling, which is more acceptable.

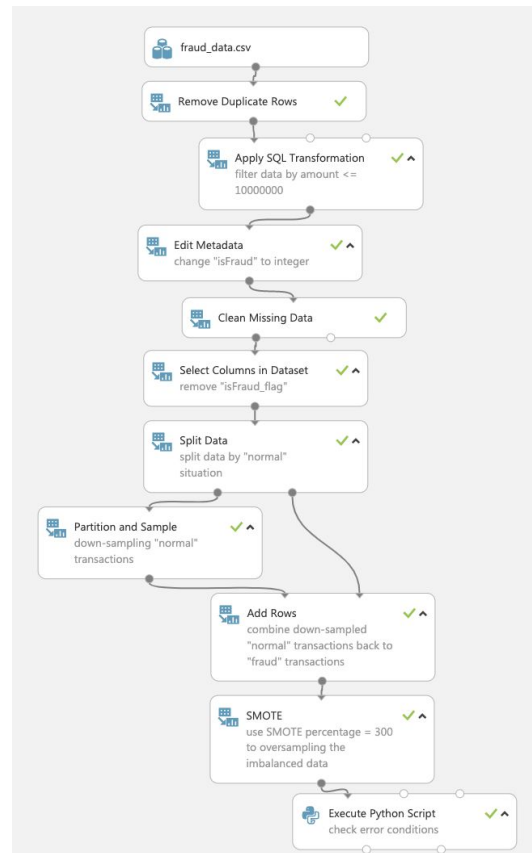


Fig 2. Data processing as the first step.

Then we conduct feature engineering on the sampled data as shown in Figure 3. We separate the columns from raw data into three groups: categorical features, features can be used to derive new features, and features to keep. We first transform categorical features into count table parameters. Then we use columns such as 'nameOrig' and 'nameDest' to form our feature set.

- 'is_Orig_Dest_same': an indicator for whether original account name is the same as the destination account name
- 'is_Orig_change': an indicator for whether 'oldbalanceOrig' is the same as 'newbalanceOrig' for each transaction
- 'is_Dest_change': an indicator for whether 'oldbalanceDest' is the same as 'newbalanceDest' for each transaction
- 'is_Dest_zero': an indicator for whether 'oldbalanceDest' is zero before the transaction
- 'is_Orig_from_zero': an indicator for whether 'newbalanceOrig' is zero for each transaction
- 'is_Dest_Merchant': an indicator for whether the destination account is merchant or not

After combining all the features, we end up with 12.

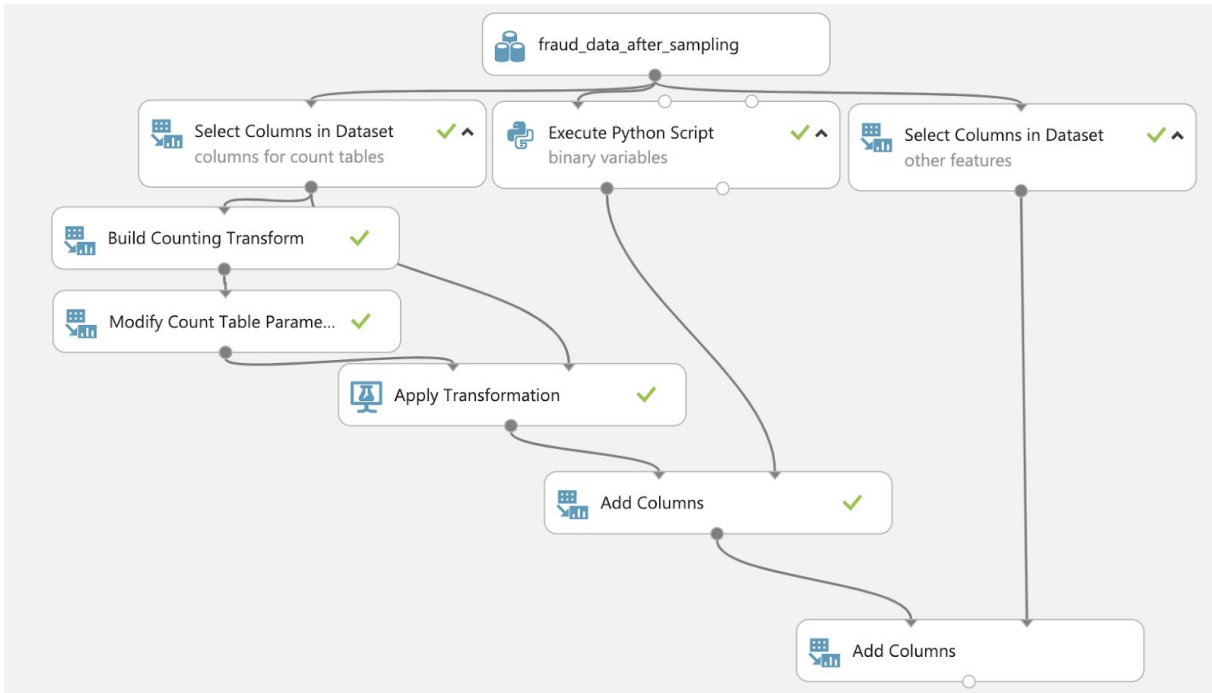


Fig 3. Feature engineering as the second step.

Then we start building the machine learning models. Thanks to Azure Machine Learning studio, we can try various models conveniently without writing any code. We split data as 75% being training data and 25% being testing data. We first tried PCA-based anomaly detection with hyperparameter tuned based on F-score. This process is shown in Figure 4 below.

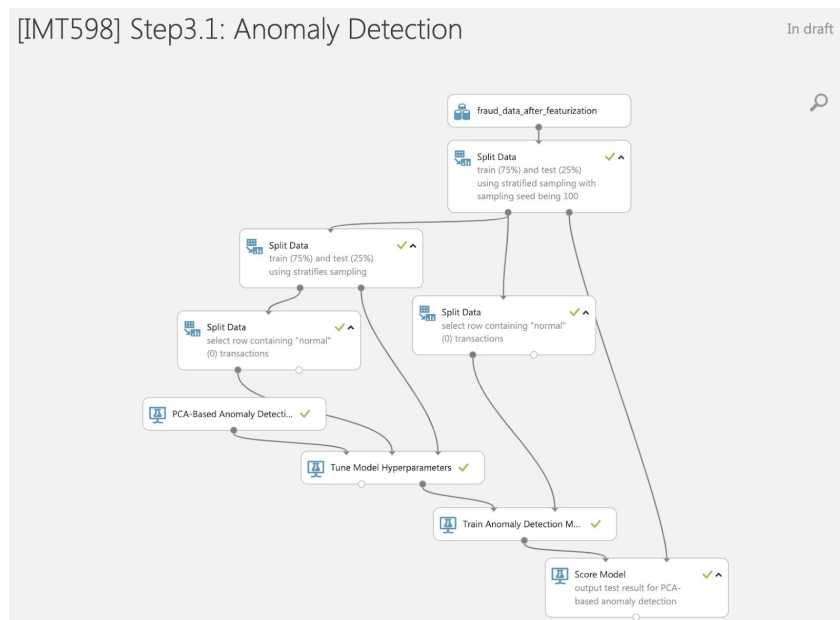


Fig 4. Anomaly detection as a part of the third step.

We also built three kinds of classification models, as shown in Figure 5. All the models have hyperparameters tuned using F-score. One model is two class logistic regression model. The regularization weights for both L1 and L2 are 1. One model is two class decision forest using bagging as resampling method. The number of decision trees is 20 and the maximum depth of decision trees is 32. The number of random splits per node is 128 and the minimum number of samples per leaf node is 5. The fourth model is a two-class neural network. The number of hidden nodes is 100. Learning rate is 0.1 and the number of learning iterations is 100. The initial learning weights diameter is 0.1.

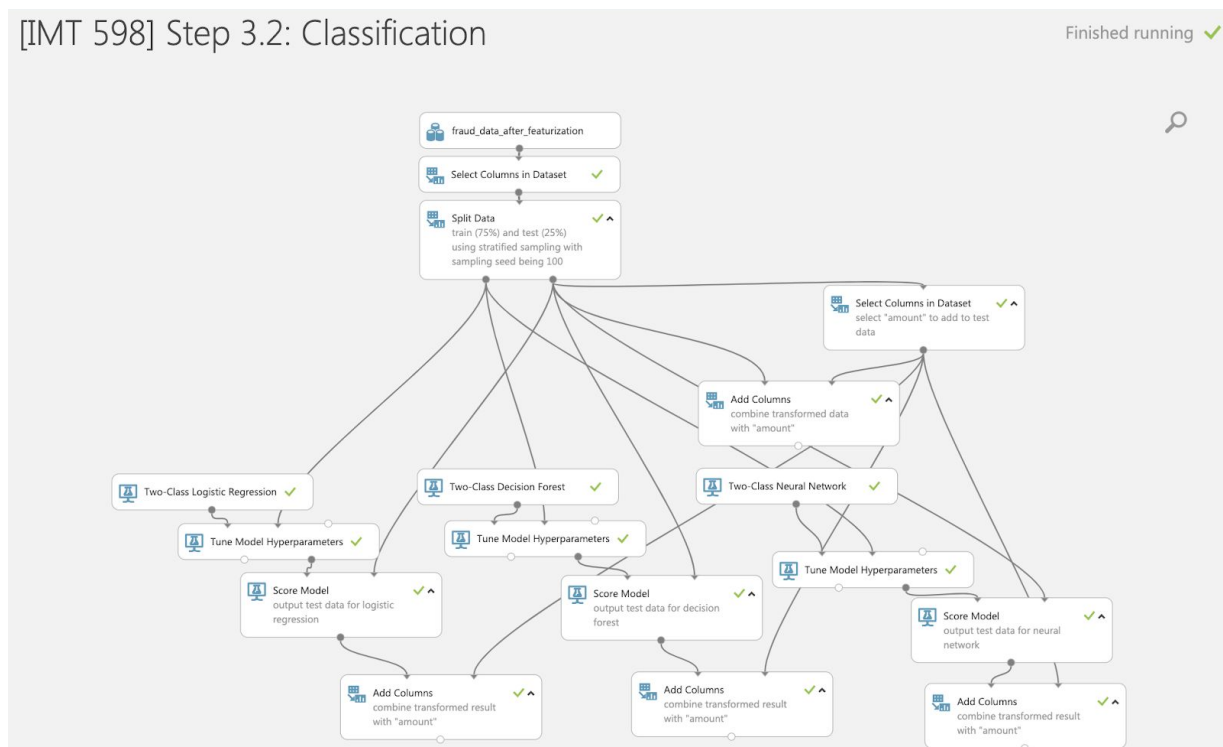


Fig 5. Classification models as a part of the third step.

After we finished training the models mentioned above, we used the split testing data to test the performance of each model and saved the testing results for the next step of model selection. As for the traditional machine learning metrics, the results are shown in Figure 6 below. It is obvious that two-class decision forest has the best performance among these traditional metrics.


Algorithm	Accuracy	Precision	Recall	F-Score	AUC	Average Log Loss	Training Log Loss
							
Anomaly Detection	0.417036	0.071811	0.721174	0.130616	0.660331	0.706956	-208.776221
Logistic Regression	0.983061	0.972701	0.741873	0.841749	0.991722	0.05444	76.222472
Decision Forest	0.99583	0.967829	0.963351	0.965585	0.998512	0.015648	93.165659
Neural Network	0.98457	0.960463	0.777913	0.859603	0.994018	0.042999	81.219275

Fig 6. Decision Forest performs best among all the traditional machine learning metrics.

As we mentioned before, we also used three specific financial related metrics ADR, VDR, and AFPR to measure the performance of each model. Based on the definitions of these metrics, ADR and VDR should be as large as possible, and AFPR should be as small as possible. Then the results show that decision forest is the best considering ADR and VDR. Logistic Regression is the best considering AFPR. However, the performance of decision forest on AFPR is similar to that of logistic regression.

Table 1. Financial related

Method	ADR	VDR	AFPR
Anomaly Detection	72.12%	81.38%	92.82%
Logistic Regression	74.19%	96.69%	2.73%
Decision Forest	96.34%	99.62%	3.23%
Neural Network	77.79%	97.75%	3.95%

Considering all the metrics, we select the trained decision forest as the model to implement on our system. We deploy the model in Azure Machine Learning studio and then connect its API to our web service developed in-house.

Web App

In order to provide an interface to our machine learning solution we created a python based Web App. We followed the instructions laid down by Cephas Lin et al (2019). The WebApp has been deployed on Azure Linux Free-Tier VM. It can be accessed on <http://fraudwatch.azurewebsites.net/>. The Web App uses Bootstrap as a front-end and Flask as a backend. It has been deployed on the Linux server through the utilities provided by Azure CLI. The HTML form interactions are implemented using the wtforms library in Python.

The Web App is an HTML form which accepts inputs and flashes the fraud probability on clicking the submit button. Out of all the fields in the data set the Web App asks for the following six features:

- Type: the type of transactions, having values [CASH-IN, CASH-OUT, DEBIT, PAYMENT, TRANSFER]
- Amount: amount of the transaction in local currency.
- nameOrig: customer who started the transaction
- nameDest: customer who is the recipient of the transaction
- oldbalanceOrig: initial balance before the transaction for origin account
- oldbalanceDest: initial balance before the transaction for destination account

The following fields are calculated by the web app by using amount, oldbalanceOrig and oldbalanceDest:

- newbalanceOrig: initial balance before the transaction for origin account
- newbalanceDest: initial balance before the transaction for destination account

After acquiring the information from the user, the Web App requests an API call from the Azure ML Studio service to fetch the probability of Fraud. The Web App is also provided with error handling and feature validation. For example, it only accepts a decimal input for amount and can handle errors with respect to connection with external API.

Fraud Watch

Transaction Type:

☒ PAYMENT ☐ TRANSFER ☐ CASH_IN ☐ CASH_OUT ☐ DEBIT

Amount:

\$

Name Origin:

Transaction Origin

Old Balance Origin:

0

Name Dest:

Transaction Destination

Old Balance Destination:

0

Submit

Results

The output for our digital transformation system is a web page providing us with the predicted fraud probability for each transaction information entered into the system. Right now the performance of the machine learning model is quite accurate with accuracy being 0.99583, precision being 0.967829 and recall being 0.963351. Therefore, we can conclude that our machine learning model applied in the new digital system is reliable. This selected machine learning model, which is a two-class decision forest mentioned above, serves as the core function is the proposed new digital transformation strategy. The digital system is designed to accept streaming data and each transaction appears to be one row in the data. For all the pending transaction, the digital system will input it into the deployed machine learning model to predict the probability of it being a fraud transaction, Our current strategy is to reject any transaction with fraud probability larger than 0.5.

KPI - Key Performance Indices for your initiative

We used the three financial related metrics ADR, VDR, and AFPR as the key performance indices to evaluate our new strategy performance. Under the rule of

triggering block action when the output (fraud probability) being larger than 0.5, the performance of our fraud detection system has a great improvement compared with the previous one (Table 2).

Table 2. KPIs improved greatly after applying the new digital transformation strategy.

Method	ADR	VDR	AFPR
New Strategy	96.34%	99.62%	3.23%
Original Method	0.19%	0.65%	0.00%

As we mentioned above, ADR indicates how successfully the new system can detect fraud transactions given all fraud ones. Therefore, we can conclude that the new digital transformation strategy performs so well taht it will catch nearly all the fraud transactions. As for VDR, which indicates the percentage of money saved after applying the new digital strategy, our new system reduce almost all the risk of losing money due to fraud transactions for Freya Group. AFPR, defined as the ratio of false detected fraud transactions among all the detected ones, should be as small as possible as it indicates the mistaken rate in fraud detection. Our current strategy has AFPR as 3.23%, which is acceptable and may still needs improvement in the future.

Discussion

The most interesting sub-problem in this project is how to conduct feature engineering for building machine learning models. We only have eight useful columns that can be considered as potential features in the raw data. Not to mention that some of them cannot be used directly in the model due to the characteristics of the data. Therefore, we brainstormed on what kind of features be generated. For example, we determined that 'is_Dest_zero' might be a useful feature since it indicates whether or not the destination account is zero before transactions, which can be a critical factor when predicting fraud. Also, we constructed a feature called 'is_Dest_Merchant' to indicate whether the type of destination account. In our data, we only have two types of destination account, customer and merchant. Therefore, we can simply create a flag to show its account type.

The most difficult decision made during this project should be how to select model. Overall, we have 10 metrics to consider when evaluating a model. Most of the time we should weigh the importance of each metric to determine which one to follow. Luckily, the only contradiction in this case exists in comparing AFPR for logistic regression and decision tree. Logistic regression has the smallest AFPR (2.73%) and that for decision tree is 3.23%, which is slightly larger. However, decision tree performs the best considering all the other 9 metrics. Therefore, we finally decided to use decision tree as its minor drawback in AFPR is negligible.

Learnings

The key learning of this experiment is noted below:

The art and science of machine learning has been greatly simplified by Azure ML services. We conducted a really complex machine learning experiment which included feature engineering, model selection and nine metrics for evaluation not to mention over a large dataset. However, the time and effort spent for this activity is much lesser as compared to implementing everything from scratch.

Productioning a code has never been easier. Azure's extensive cloud stack as the ease at which the code could be deployed on a server helped us deploying the app and make it publically available in minutes. We did run into challenges with respect to integrating the ML Studio with the Web App. However, those challenges were nothing compared to the issues we would have faced had we use an on-premise server on WAMP for example.

Assumptions

We assume that all the transactions will stream into the system immediately and the time for predicting whether an incoming transaction is fraud or not can be negligible, which means that we have time to respond to fraud ones.

Future Work

Due to the lack of temporal data, we did not conduct time series analysis, which might be helpful to our model improvement. Therefore, we expect to collect more data during future operation and seek to improve the current machine learning model. Also, we want to figure out the best probability threshold to flag a transaction as fraud. Currently, we use 0.5 to trigger the block operation. However, we may want to introduce a more gentle way to operate the business. For example, we may only trigger the block operation when the fraud probability is larger than 0.7. As for the transactions predicted to have a fraud probability between 0.5 to 0.7,

we may want to contact customers for verification first before conducting any operation directly from the bank side.

Related Work or Research

Tamil Bharathi. (2019, February 25). How is Digital Banking Transforming Fraud Detection?. Retrieved from <https://www.zucisystems.com/how-is-digital-banking-transforming-fraud-detection/>

Somesh Khanna, & Heitor Martins. (2018, April). Six digital growth strategies for banks. Retrieved from <https://www.mckinsey.com/business-functions/mckinsey-digital/our-insights/six-digital-growth-strategies-for-banks>

AzureML Team. (2015, March 18). Online Fraud Detection. Retrieved from <https://gallery.azure.ai/Experiment/66e6ca668a5041058f388d35730c05af>

Yuzhong Song. (2015, December 18). Online Fraud Detection: working with unbalanced class data. Retrieved from <https://gallery.azure.ai/Experiment/Online-Fraud-Detection-working-with-unbalanced-class-data-1>

Cephas Lin et al. (2019, October 22). Quickstart: Create a Linux Python app - Azure App Service. Retrieved December 7, 2019, from <https://docs.microsoft.com/en-us/azure/app-service/containers/quickstart-python?tabs=bash>