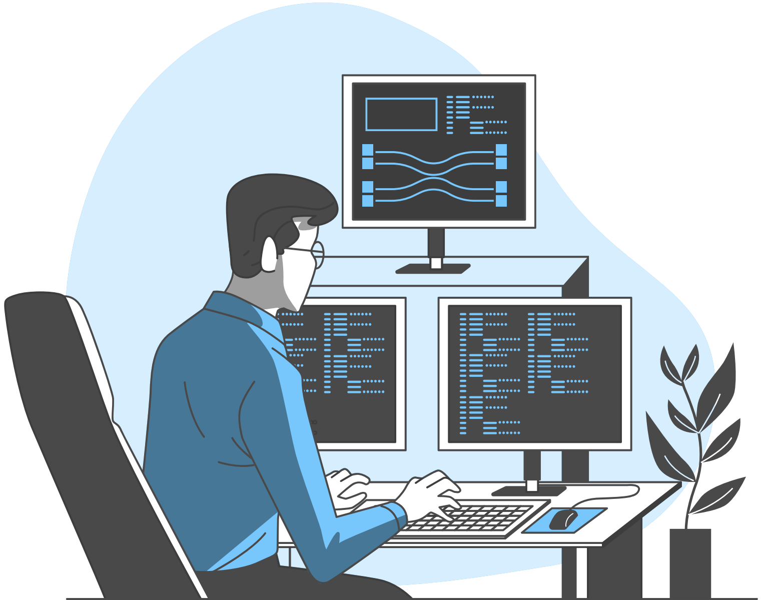
The American Express Innovation Labs AI Hackathon (Singapore) 2024

Team AInnovative



Problem Statement & Uplift Model

Problem Statement:

“Maximize **Incremental Activations**, activations on merchants that the Customer would not have discovered otherwise, unless **recommended**. “

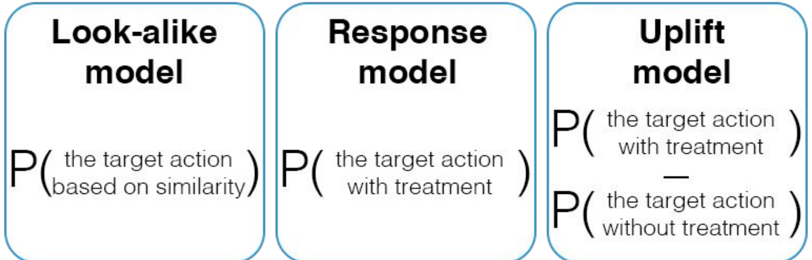
Why Uplift Model is suitable:

Focuses on Causal Impact: Uplift models estimate the causal effect of a treatment (in this case, the recommendation) on an outcome (the customer activation). This means it isolates the impact of the recommendation, removing the influence of factors that might have led the customer to the merchant anyway.

Compares Treated vs. Control Groups: Uplift models typically compare two groups: one group that receives the treatment (the recommendation) and a control group that doesn't. By analyzing the difference in activation rates between these groups, the model estimates the incremental effect of the recommendation.

Accounts for Selection Bias: Uplift models can address selection bias, which occurs when customers who are more likely to activate with a merchant are also more likely to receive a recommendation for it. Uplift models help isolate the true effect of the recommendation, not just the selection bias.

Uplift Model is different from response model and look-alike model



Our Uplift Model

1. Import library

```python
import pandas as pd
from sklearn.ensemble import RandomForestClassifier, RandomForestRegressor
from causalml.inference.meta import BaseXClassifier
from imblearn.over_sampling import SMOTE
seed = 1
```

2. Import dataset as Pandas dataframe

```python
train_set = pd.read_csv(r'/Users/marcus/Documents/ae hackathon/dataset/65d4f0fcb8af9_amex_campus_challenge_train_3.csv')
eval_set = pd.read_csv('/Users/marcus/Documents/ae hackathon/dataset/65d4b8b2ebfe9_amex_campus_challenge_eval_round1_2.csv' )
```

3. **Fill missing value** and prepare two dataframe

```python
train_set.fillna(0.0, inplace=True)
```

```python
# Move 'distance_05' column to the first position for train_set_recommended
if('distance_05' in train_set.columns):
    col = train_set.pop('distance_05')
    train_set.insert(0, col.name, col)

train_set_recommended = train_set[train_set['ind_recommended'] == 1].drop(columns=['customer', 'merchant'],axis=1,inplace=False)
train_set_not_recommended = train_set[train_set['ind_recommended'] == 0].drop(columns=[ 'customer', 'merchant'],axis=1,inplace=False)
```

4. Use **SMOTE** to sample train data

```python
def balance_classes_with_smote(df, class_column, sample_size, seed, k_neighbors=5):
    smote = SMOTE(sampling_strategy='auto', k_neighbors=k_neighbors)

    # Sample rows where activation is 0 and where activation is 1
    filtered_non_activation = df[df[class_column] == 0].sample(n=sample_size, random_state=seed)
    filtered_activation = df[df[class_column] == 1]

    # Concatenate the filtered dataframes
    df_balanced = pd.concat([filtered_non_activation, filtered_activation], axis=0)

    # Apply SMOTE to the activation data
    X_resampled, y_resampled = smote.fit_resample(df_balanced.drop([class_column], axis=1), df_balanced[class_column])

    # Combine the resampled data
    df_resampled = pd.DataFrame(X_resampled, columns=df_balanced.drop([class_column], axis=1).columns)
    df_resampled[class_column] = y_resampled

    return df_resampled

def balance_classes(df, class_column, sample_size, seed):
    filtered_non_activation = df[df[class_column] == 0].sample(n=sample_size, random_state=seed)
    filtered_activation = df[df[class_column] == 1].sample(n=sample_size, random_state=seed)

    return pd.concat([filtered_non_activation, filtered_activation], axis=0)


# Apply balance_classes_with_smote function to train_set_recommended
train_set_recommended = balance_classes_with_smote(train_set_recommended, 'activation', 30000, seed)

# Apply balance_classes_with_smote function to train_set_not_recommended
train_set_not_recommended = balance_classes(train_set_not_recommended, 'activation', 30000, seed)
```

```python
train_total = pd.concat([train_set_recommended,train_set_not_recommended],axis=0).sample(frac=1,random_state=seed)
```

5. Use X learner with **outcome learner = Random Forest Classifier** and **effect learner = Random Forest Regressor** to fit train set and predict evaluation set

```python
x_learner = BaseXClassifier(outcome_learner=RandomForestClassifier(n_estimators=100,random_state=seed)
                            , effect_learner=RandomForestRegressor(n_estimators=100,random_state=seed))

x_learner.fit(
    train_total.drop(columns=['activation','ind_recommended'],axis=1,inplace=False).values,
    treatment=train_total['ind_recommended'].values,
    y = train_total['activation'].values
)
```

```python
y = x_learner.predict(eval_set.fillna(0).drop(columns=['merchant', 'customer'],axis=1,inplace=False).values)
submission = eval_set[["customer","merchant"]].assign(predicted_score=y)
```

```python
submission.to_csv('submission.csv',index=False)
```

Dataset

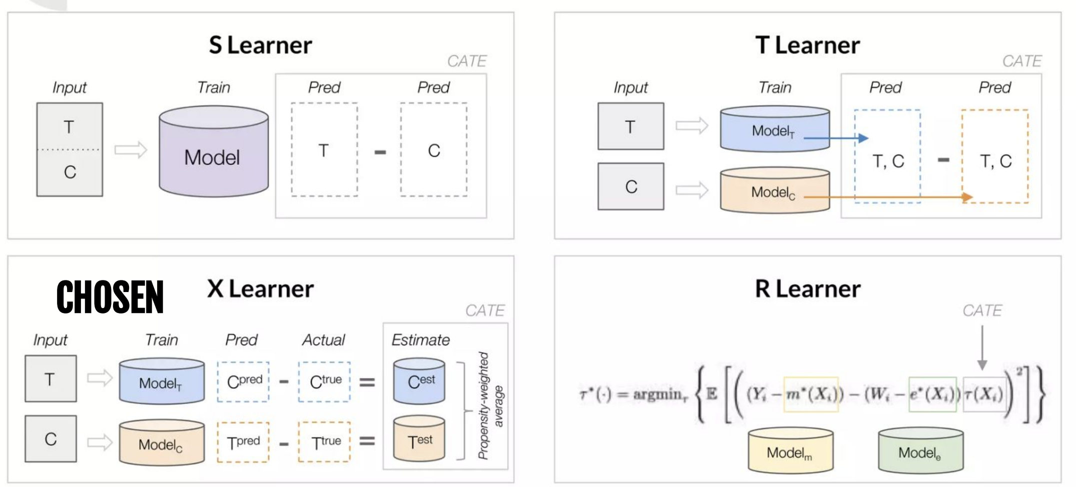
Issue	Description	Negative effect on modeling
Features with > 50% missing values	e.g. merchant_spend_11, customer_digital_activity	missing values can lead to a loss of information, which is critical for accurately predicting uplift. Techniques to handle missing data, like imputation, can introduce bias or distort the true underlying relationships in the data.
Mixed aggregation level dataset	features are come from customer, merchant, customer & merchant, industry level	features from different levels of aggregation may not align properly, leading to a mismatch in the scale and granularity of the data. This can cause the model to misinterpret the effects of the treatment across different levels.
High Variance & outlier	e.g. customer_spend	high variance can make the model less stable, and outliers can skew the results
Highly correlated features	e.g. customer_spend, customer_digital_activity	highly correlated features can cause multicollinearity, which makes it difficult to determine the individual effect of each feature on the response variable.
Unbalanced class & lack of minority class data	activated and recommended data only have 10k rows, account for 0.08% of all data	an unbalanced dataset can lead to a model that is biased towards the majority class, resulting in poor classification performance on the minority class.

Data Processing

Data preprocessing method	Description	Problem Addressed	Advantage	Disadvantage	Chosen ?
Filling missing value	**CHOSEN** Fill missing value with 0	Missing value	Quick and easy to implement	Missing value may have their own meaning. Introduce bias	✅
	Down sample		Can help balance the dataset; reduces computational load	The sample is too small, may fail to capture the true variance of data	
	Duplicate minority class data 3 times		Better representation of minority class. Enlarge the training dataset	Can lead to overfitting to minority label data	
Down sampling	**CHOSEN** Use SMOTE to create minority class data	Unbalanced Class	Generates synthetic samples; improves generalization	Can introduce noise; synthetic samples may not be representative	✅
	Min max standardization		Puts all features on the same scale; easy to understand	Sensitive to outliers; does not handle skewed data well	Standardization and PCA are avoided due to observed overfitting. When the training dataset lacks representativeness , applying these methods to further reduce variance can exacerbate overfitting.
	Normalization				
	Winsorization + Normalization	High Variance & outlier	Ensures features contribute equally to distance measures	Can be distorted by outliers; assumes linear relationships	
Standardization	Normalization each feature according to their dimensional mean and standard deviation	Mixed aggregation level dataset	Ensure features are scaled according to their own mean and sd	Heavy workload	
Dimensionality reduction	Normalization + PCA	Highly correlated features	Improves computational efficiency; reduces overfitting;	Feature selection can discard important information; may lose significant variance	

Uplift Modeling Meta Learner

Model	Brief Summary
S-Learner	Uses a single model to predict intervention effects across the entire dataset.
T-Learner	Trains separate models for treatment and control groups, estimating intervention effects by comparing predictions.
R-Learner	Focuses on predicting individual-level treatment effects using estimation residuals as targets.
X-Learner ✅ **CHOSEN**	Uses estimates from treatment and control groups to improve predictions, especially for minority groups, by crossing these



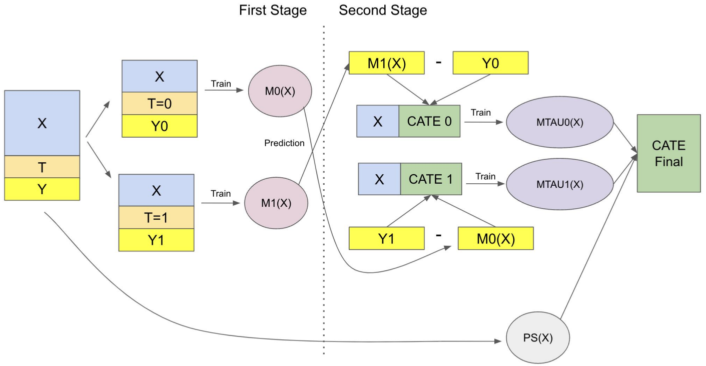
Evaluation Metrics

Evaluation Metrics	**Description**	**Advantage**	**Disadvantage**
Top 10 Incremental activation rate	The difference in activation rate between the treatment and control groups at specific quantiles (percentiles) of the predicted uplift score.	- Easy to understand and communicate to stakeholders. - Identifies segments with high potential uplift, allowing for targeted treatment strategies.	- Limited view of overall model performance, as it only focuses on specific portions of the uplift distribution. - Ignores information from lower quantiles of the data, which may still contain valuable insights.
Area Under the Uplift Curve (AUUC)	Measures the entire cumulative uplift across all quantiles of the predicted uplift score. The AUUC value ranges from 0 (no uplift) to 1 (perfect uplift).	- Captures the overall performance of the model for uplift modeling. - Considers the uplift potential for all users, not just those at the top.	- Doesn't reveal the distribution of uplift scores, making it difficult to assess model behavior across different user segments or set optimal treatment thresholds.
Uplift Score Distribution	Analyzes the distribution of predicted uplift score of user. This can be visualized using a histogram or density plot.	- Helps set optimal treatment thresholds by identifying the portion of the user base with the most promising uplift potential. - Provides insights into model behavior across different user segments, as the distribution can reveal patterns or biases.	- Limited standalone performance metric, as it doesn't directly measure the effectiveness of the model in achieving the desired outcome (e.g., increased activation rate).

By utilizing a combination of these metrics and considerations, we can gain a comprehensive understanding of our uplift model's performance and make informed decisions about its deployment and optimization.

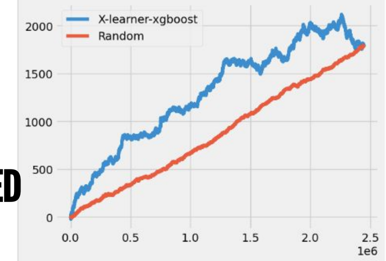
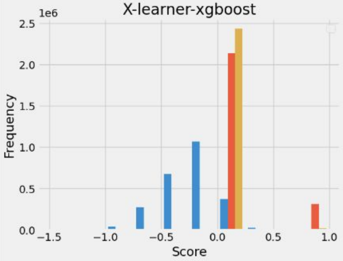
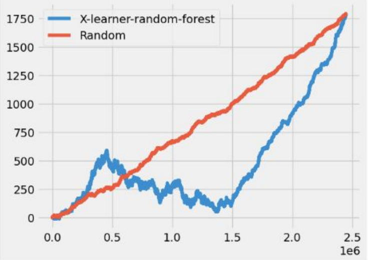
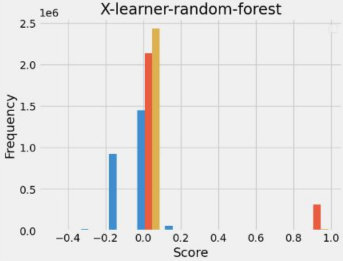
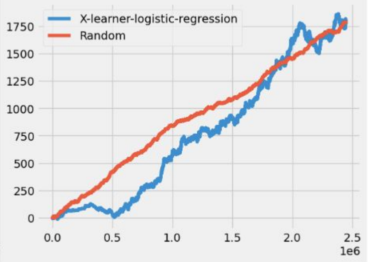
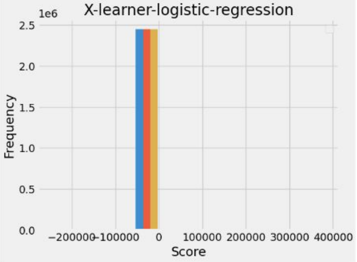
Why We Choose X-learner

Why X-learner	**Comparison**
Improved Estimation of Heterogeneous Treatment Effects	While **T-learner** leverages separate models for treatment and control groups, it might not fully utilize shared information between them. This can lead to suboptimal estimation of treatment effects, particularly when effects vary across individuals.
	S-learner method utilizes a single model to predict treatment effects for both groups. However, in scenarios with strong heterogeneity (effects differ significantly across individuals), S-learner might struggle to capture these nuances, leading to inaccurate estimations.
	R-learner often requires complex tuning and optimization processes. This can be time-consuming and computationally expensive.
Scenarios with Data Imbalance and Sparse Treatment Group Data	**X-learner** demonstrates robustness when datasets where the control and treatment groups have unequal sizes
	T-Learner: When the treatment group data is scarce, T-learner's reliance solely on that data for estimation can be problematic. The lack of sufficient information can lead to inaccurate treatment effect estimates.



High Level idea of X-learner

Classification Model Comparison

Model	AUUC	Uplift Lift Score Distribution	Top 10 Incremental Activation Score
RECOMMENDED XGBoost			0.00077
CHOSEN Random Forest			0.00074
Logistic Regress			0.00073

- X-learner is used. All models use their default parameter value.

- **Random Forest is chosen for evaluation, but XGBoost is a better choice.**

- **Random Forest** achieve the highest Top 10 incremental activation score in the evaluation submission.

- **Random Forest** performance is concentrated on user-merchant pairs with already high uplift scores, as shown by the AUUC graph. This suggests overfitting to specific cases.

- **XGBoost** is more robust. It outperforms random selection across all user-merchant pairs, demonstrating a more generalizable uplift prediction. Additionally, XGBoost's output exhibits a more normal distribution of uplift scores, indicating a more balanced prediction across the data.

Next Steps

Technical Enhancements:

- **Advanced Uplift Modeling:** Explore DragonNet for uplift modeling, Random Forest Uplift modeling, etc. leveraging their ability to handle complex patterns in high-dimensional data efficiently.
- **Classification Model Parameter Tuning:** Improve the prediction accuracy by parameter tuning.
- **SHAP Value Interpretation:** Utilize SHAP value calculation to understand feature importance and enhance model interpretability.
- **Sensitivity Analysis**: Test model robustness with techniques like placebo treatment, irrelevant confounder introduction, and subset validation.

Business Validation:

- **A/B Testing**: Conduct A/B testing with **random selection** (control group) and **high uplift score user** (treatment group).
- Metrics: Evaluate conversion rate, average spending, and user retention to assess the uplift model's impact.