bullet point

Wednesday, June 19, 2019

9:32 AM

Note:

Resume上的project心里要过一遍。 要让interview你的人感觉你做的project是可以scale的，不是那种小的ad-hoc的。

Brief intro:

I currently worked in risk modeling team at Santander Bank for 2.5 years. My main responsibility is to develop and implement **statistical models** to forecast the credit loss for the bank's commercial and consumer portfolios. I used **statistical models/machine learning methods** like logistic regression/decision tree to predict the loss rates for these loans. For coding and analysis, I mainly use python, SAS and SQL.

Because the users of our model are partners of different business lines, they use our model to forecast credit loss for **strategy planning** and also **stress testing** for the bank, so I work closely with business partners for the **full cycle** of the model building process. Meaning we communicate frequently with the business partner about the business requirement and model methodology. So I believe I have sufficient **knowledge of the financial** market and financial products that is related to this position (**products of our bank**) and also have strong ability to solve the business problem using analytical tools.

I also work closely with IT department to gather and validate the required data elements for model development. I believe I also have strong **programming and data manipulation skills**.

* • Developed and implemented statistical loss forecasting models to predict PD, LGD, and EAD using logistic regression model in Python and SAS for Bank's $30 billion commercial portfolio.

For C&I model: Go to C&I-C&I PD LGD LEQ model tab

For Streamlined model: Go to Streamlined Model MDD!!! tab

* **Goal**:
* First I will start with a little background of what I am doing for my current job. The goal for a bank is to maximize the rate of return by maintaining an acceptable risk level. So It is crucial for the bank to forecast the revenue and risk for the next few years so that it can do its strategy planning based on the forecast. My team is under the risk modelling group so the main work for our team is to predict an accurate and reasonable level of potential loss for the bank's $100 billion portfolio. As Santander Bank is a retail and commercial bank, the most of the risk is credit risk which comes from when the bank borrower fail to meet their obligations to pay back the money.
* Predict accurate loss rates; too high: set aside too much money, waste; too low: not enough reserve for potential risk, Fed will fine;
* **Approach**:
* When I first joined the team, the performance of existing model is not very well. The two biggest challengers I am facing is that a: data is so messy; b: the model used is not complex enough to capture the risk characteristic of portfolio
  1. **Data** messy: aggregate all kinds of source of data, bridge between business and IT;
  2. Data needed for the model execution is from different databases. The existing process is that for each model run, we gather the data from different databases and process, merge them together as model input. This will can cause a problem that the refreshed frequency and refreshed date of different database can vary a lot, meaning the result of running the model from yesterdays can be different than the results from today.
  3. *Also, we need to manually download the data from several databases into excel spreadsheet as the reference files for model execution. This can cause a lot of problems. Firstly the data value and format can go wrong when dump the data in spreadsheet. For example the precision of numbers can change, or the format of data element can change (character format change to numeric ).*
  4. My proposed process is that first of all we **aggregate** all the data needed for model execution into one single **centralized database**, this will make sure all the data are consistent and in the correct format. To do that we need to merge multiple data together using proper business logics. So I work with business team to figure out all the correct logics and relationship between these data. For example when I merge financial statement data, I need to work with business team to figure out which statement should be put in as the most current statement.
  5. Then I propose to use python for **preprocessing** and **validation** of the data as it provide plenty of packages and methods that are easily used to clean data:
     + Impute the missing data: Example: impute FICO score: different source/transformation/
     + implement the correct logic: Example: get the correct financial statements (filter for annual statement/valid audit method filter and rank/use the latest statement)
  6. **Model** is too simple to capture the risk characteristic of the portfolio; build logistic regression mode
  + The existing model is a simple risk scorecard developed several years ago. It is more like an expert judgement process instead of a statistical model.
  + We hired Mckinsey to develop to framework of our model, we recalibrate the model as we have more historical data coming in.
  + **Methodology** **choice**:

We tried methodology like logistic regression, decision trees, clustering, and other machine learning methods. We compare factors like prediction accuracy and ease of interpretability. Since the model is reviewed by regulators like Fed/OCC, the ease to interpret to the audience is very important. We also found that some machine learning methods like decision tree did not result in significantly superior forecasts in the sense of the forecast accuracy. So the regression models were, therefore, selected as the final models because of their advantage in **transparency** and ease of **interpretability.**

We used decision tree model as **benchmark** model to compare the predict accuracy.

* **Variable choice**:
  + **Loan** features: Location/Loan type/; commercial: financials; retail: FICO scores
  + **Macroeconomic** Variable (required to be sensitive to the macroeconomic): Use the **original** and **transformation** of the macroeconomic variables to test the significant values
    - Commercial **C&I** (financials: EBITDA/Quick Ratio/DebtToTNW/Profit margin/Sales growth/Current statement; LGD variable: collateral type (commercial property/financial property)/Santander Risk Rating (threshold of 4)/facility type (Revolving line of credit/Term loan); + macro economics: Unemployment rate/HPI/GDP/10 year bond spread)
    - **CRE**: LTV/Occupancy rate(percentage of occupied units in a commercial property)/Month to maturity + macro economics: Unemployment rate/HPI/CREPI(CRE price index); Add from DebtX: Collateral type/Debt service coverage(net income/interest&principle payment)/Interest rate type
    - Consumer (**Small Business Banking**: FICO/Location/Product type (CRE/CNI, Line/Term)/SBA (guaranteed by Small Business Administration)/SIC(Standard Industrial Classificatiob: industry concentration) + macro economics: Unemployment rate/Salary growth/household obligation ratio);
    - For **mortgage**, add Product type is (fixed/floating rates)/LTV/Loan Age/Jumbo loan + macro economics: Unemployment rate/HPI/1 year LIBOR rate; For **HELOC**, add first/second lien

* **Result**:
  + Improve the accuracy
  + Pipeline I created in python has become the standardized method our whole team to use;

* Developed a Python **Model-View-Controller (MVC) framework** for model implementation that reduced implementation time from 2 to 3 months to two weeks, increased robustness through extensive unit testing, input parameter validation and version control and reduced execution time by approximately 80%.
* **Goal:**
* There are three main portions of our program, the portion of connecting to database and clean the data, the portion of predictive model, and the portion of showing some graph about the input and output. We want to use a framework that separates these portions of codes so that we won't touch the code of other portions if we only need to refresh part of the code.

**Approach:**

* I did some research about some framework widely used in the industry and found that the MVC framework can do exactly the thing we want.
  + **Model**: part of code that **manage** the data including **connection** to the external database to get loan data and macroeconomic data and other model inputs, and **process/clean** the data so that it can be ready for model to use
  + **Controller**: part of code that actually **execute the business logic** including like the model **calculation**, generate the model **output**. This is where all the model coming in like regression/decision trees.
  + **View**: part of code where the model user actually **interact with the system** including like user log-in and select the model and data to run and generate model output and results to show. This also include tools that can be used for data visualization easily like showing a chart of some key features of the input data, and some charts of model output rates.
* Detail:

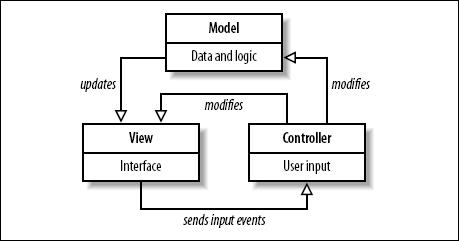
The python implementation framework architecture adopted is **Model, View, and Controller (MVC):**

**Model**: Manages the **data**, data logics, and processes and connections to external **databases**. This includes class agents in charge of providing **portfolio data**, **macroeconomic scenarios**, and other data **input** requirements, such as model input parameters or **model properties**.

**Controller**: Accepts inputs and converts it to commands and **executes** the business logic (the actual stress testing models and EJMs). Here is where the **actual model calculations** happen. It includes the model classes, **contributor file** utilities, and other **utilities**, such as logging, session management, and generic utility functions.

**View**: This is any output **representation of information**, such as **charts, tables, or input forms**. The view is where the **human interacts** with the system. This includes the framework’s web interface where the user logs in and is able to select and run models, produce, and download contributor files and audit logs, as well as visualize selected output metrics.

Figure 68: Process Flow Diagram – MVC



After executing the models, the output of this system is a **contributor dataset containing all the values** for all metrics required by the SAS Aggregation Engine. In addition to the contributor data, the framework also create a detailed **process log** that records all steps executed within the process.

**Result:**

* Pipeline/framework I created in python has become the standardized method our whole team to use;

* • Visualized the key risk drivers and performed sensitivity analysis for model review, and put together model performance reports for management and external audience including Federal Reserve and OCC.

Key **risk driver** plots:

We will monitor some of the key risk drivers for different products. Take the loan feature above as example. We use **bar chart or pie chart** to show the distribution of these key risk drivers.

When the distribution of the key driver change a lot, we will take it very seriously since it may means that some of the assumption of previous model has change. We then will work with business to decide whether we need to recalibrate the model.

**Sensitivity** analysis:

We conduct sensitivity analysis of some important macro-economic variables. Business team want us to show them the contribution of each variables to the final results. For example, after we build a regression model, although we can tell the rough effect of each variable by checking the coefficients of the regression model, but that is hard to **quantify the contribution**. So we will plus or minus one variable by its standard deviation and fix all the other variables, and we re-run the model to see the updated results. We do this for all variables and create the **pie chart** to show the contribution of each variable.

Also, if we run several scenarios like base/adverse/severely adverse and have the loss numbers for these three scenarios, and we want to know that the contribution of all scenarios to the increment loss from base to adverse. Then we can try change one scenario to adverse while holding all other scenarios as base, run the model and get the loss numbers. We do this for each variable and we can use **waterfall plot** to show the contribution.

Use **Tableau**:

We usually use python or SAS to create plots as python; Many package can be used in python like Matplotlib/ggplot/seaborn;

But if I want to analyze your data **geographically**, I will plot the data on a map in Tableau. For example, I want to show the loss rates forecast for different regions, **a map view** is the best visualization tool for the data. The audience can see very clearly about the which state has higher loss rates, which state has more customers. For those states with low loss rates but fewer customer, meaning it may has huge potential for that region, we will recommend to the business departments asking them to target more customers there.

* • Coordinated closely with model developers and business department to discuss model methodologies and results, and worked with IT department to gather and validate necessary inputs for model execution.
* Experience working with business/IT

Talk technical with business; whole cycle educate them (logistic regression: coefficients)

Talk business with IT; educate them (preprocess the financial statements; calculate the ratios)

Working with **business**:

Because the users of our model are partners of different business lines, they use our model to forecast credit loss for **strategy planning** and also **stress testing** for the bank, so I work closely with business partners for the **full cycle** of the model building process. Meaning we communicate frequently with the business partner about the business requirement and model methodology.

* Firstly, we will discuss about **business requirement**. Make sure we fully understand the business **background** (for streamlined: need a new loan-level model to replace current top-of-the-house model, provides estimation at a more granular level than the existing model), purpose of the model (strategy planning and stress testing), and how they want to use the model.

* Then, we will get some insights of **key data elements** needed for model development from them. Since they have better understanding of the business and customer, they will provide us a list of key elements they suggested we should be looking at. There are two sets of data: (detail see above)
  + **Customer level data**: 100+ total, 20+ focus;
  + **Macro-Economic** indicators: Fed require the forecast to be sensitive to the macroeconomic variable; 20+

* After doing some preliminary analysis of data, we will propose **model methodology** to business depending on the model purpose and available data.

We tried methodology like logistic regression, decision trees, clustering, and other machine learning methods. We compare factors like prediction accuracy and ease of interpretability. Since the model is reviewed by regulators like Fed/OCC, the ease to interpret to the audience is very important. We also found that some machine learning methods like decision tree did not result in significantly superior forecasts in the sense of the forecast accuracy. So the regression models were, therefore, selected as the final models because of their advantage in **transparency** and ease of **interpretability.**

We used decision tree model as **benchmark** model to compare the predict accuracy.

* Lastly, we will share the **model results** with the business and ask for sign-off on the model methodology and results.

Challenges:

* Explain technical terms for non-tech audience:
  + Explain logistic regression:
  + Explain the model methodology of ML: not go into the details, just explain the **basic ideas**; see B Q: Explain ML(random forest) to non-technical audience :
  + Explain model results: educate them with some basic statistical results: Logistic regression as example
    - Estimated **coefficients**: larger than 0 represent higher chances of defaulting;
    - Or: **Odds ratios** of the estimated coefficients: odds(success) = success/failure; larger than 1 represent higher chances of defaulting (the event is more likely to occur as the predictor increases);

Logit(success) = log(odds(success)) = log(success/failure); logit(success) = a + bx

* **VIF** (Variance Inflation Factor): measure the presence of **multi-collinearity** in the model; VIF < 5 means absence of strong multi-collinearity
* **P value**: coefficient significant level; see p value.

* Bridge the gap between business and IT

Work with IT:

* I also work closely with IT department to gather and validate the required data elements for model development. I believe I also have strong **programming and data manipulation skills**.

* Bridge between business and IT; aggregate different source of data into one single master database; implement the logic from business and share and teach IT
* We works with IT department to gather the required data elements for model development.
* Check availability and data quality:
  + FICO: compare different source; impute if missing
  + LTV (loan to value ratio: divide money borrowed by the value of the property): important but missing; inform and help IT to improve

* **Data** messy: aggregate all kinds of source of data, bridge between business and IT;
* Data needed for the model execution is from different databases. The existing process is that for each model run, we gather the data from different databases and process, merge them together as model input. This will can cause a problem that the refreshed frequency and refreshed date of different database can vary a lot, meaning the result of running the model from yesterdays can be different than the results from today.
* *Also, we need to manually download the data from several databases into excel spreadsheet as the reference files for model execution. This can cause a lot of problems. Firstly the data value and format can go wrong when dump the data in spreadsheet. For example the precision of numbers can change, or the format of data element can change (character format change to numeric ).*
* My proposed process is that first of all we **aggregate** all the data needed for model execution into one single **centralized database**, this will make sure all the data are in a consistent. To do that we need to merge multiple data together using proper business logics. So I work with business team to figure out all the correct logics and relationship between these data. For example when I merge financial statement data, I need to work with business team to figure out which statement should be put in as the most current statement.
* Then I propose to use python for **preprocessing** and **validation** of the data as it provide plenty of packages and methods that are easily used to clean data:
  + Impute the missing data: Example: impute FICO score: different source/transformation/
  + implement the correct logic: Example: get the correct financial statements (filter for annual statement/valid audit method filter and rank/use the latest statement)

* • Conducted User Acceptance Testing (UAT) for implementation results with multiple stakeholders to make sure its compliance with the business requirements.

* Developed and implemented statistical loss forecasting models to predict PD, LGD, and EAD using logistic regression model in Python and SAS for Bank's $30 billion commercial portfolio
* Developed a Python Model-View-Controller (MVC) framework for model implementation that reduced implementation time from 2 to 3 months to two weeks, increased robustness through extensive unit testing, input parameter validation and version control and reduced execution time by approximately 80%
* Migrated the loss forecasting model from Excel VBA to Python and SAS platform, and executed CCAR/DFAST stress tests under Federal Reserve’s regulation on a semi-annual basis
* Visualized the key risk drivers and performed sensitivity analysis for model review, and put together model performance reports for management and external audience including Federal Reserve and OCC
* Coordinated closely with model developers and business department to discuss model methodologies and results, and worked with IT department to gather and validate necessary inputs for model execution
* Conducted User Acceptance Testing (UAT) for implementation results with multiple stakeholders to make sure its compliance with the business requirements