Analyzing Failed FDIC Banks

December 15, 2017

1 Introduction and Background

Most major banks in the United States are insured by the FDIC (Federal Deposit Insurance Corporation), which, in essence, requires that institution pays a premium and adheres to specific guidelines (including quarterly reports), ensuring that the bank is run by proper business principles. In return, banks are insured which shows the banks customers that their money is safe. Banks, like any other business, can fail or go out of business. Many things may lead to these kinds of failure, including poor management decisions, or poor economic health of the nation.

When an FDIC insured bank fails, the FDIC immediately assumes all of their assets and tries to find a different bank to absorb their accounts. Usually an acquiring institution will be offered a discount rate in order to incentivize them to acquire the failing banks accounts. As the insurer, the FDIC then pays the difference in order to settle the accounts and guarantee that both depositors into the bank and loaners to the bank receive just compensation. The FDIC states: "No depositor has ever lost a penny of insured deposits since the FDIC was created in 1933".

Most of the money that the FDIC pays comes from the premiums that banks pay, but during a crisis such as the bank failure of 2008, the cost of banks failing could exceed the funds available to the FDIC. Therefore, it is critical to understand what causes bank failure and to be able to identify banks at risk by using metrics available from the FDIC.

The datasets that we will be using are all obtained from the FDIC and are publicly available at https://www.fdic.gov. The first dataset is a collection of banks that have failed since October 2000. There are 554 listed banks, 65% of which have balance sheet summaries from the time of failure available. The second dataset is a time series dataset that provides quarterly reports for financial institutions from 1992 to the present. A few key documents among these reports are the "Asset and Liabilities" and the "Incomes and Expenses" reports.

The FDIC's Center for Financial Research does research on financial institutions. One article from March 2006 titled *Trouble Banks: Why Don't They All Fail?* highlights previous research done for the same reports that we are using, but no information is readily available for any calculations determined in the past 10 years. For this reason we are confident that our analysis is possible, and that our analysis can prove to be useful by either confirming previous conclusions, or identifying how the past 10 years have affected predictors.

Throughout the project, we will refer to the banks as being in one of two main categories: failed banks and healthy banks. By "failed banks", we mean banks that have already failed. On the other hand, banks that we call "healthy banks" are not necessarily healthy. We use this term only to indicate that these banks that have not failed up to the time of this research.

1.0.1 Data Scraping, Cleaning, and Engineering

In [1]:
```
import pandas as pd
import scrape_bank # Custom file made for project
import clean_and_engineer as cne # Custom file
from matplotlib import pyplot as plt
from datetime import datetime
%matplotlib inline

plt.style.use("ggplot")                       # Use a different style.
plt.rcParams["figure.figsize"] = [6.0, 4.0]   # Make figures larger by default.
plt.rcParams["figure.dpi"] = 200              # Raise figure quality within the notebook.
```

In [2]:
```
# Warning: Scraping data takes a long time to run (~2 hours each).
scrape_failed_banks = False
if scrape_failed_banks is True:
    raw_data = scrape_bank.grab_failed_data()
    failed_df = pd.DataFrame(raw_data)
else:
    failed_df = pd.read_csv("scraped_bank_data.csv", index_col = 0)

failed_df = cne.clean_failed_banks(failed_df)
failed_df = cne.engineer_failed_banks(failed_df)

failed_summary = pd.read_csv("failed-banklist-ordered-by-cert.csv")
failed_cert_list = list(failed_summary.CERT.values)
failed_summary.index = failed_summary.CERT
form = "%d-%b-%y"
failed_summary = cne.convert_cols_to_dates(failed_summary, ['Closing Date'], form)

# This will save ~100 .zip files of about 50 MB each.
scrape_quarter_rpts = False
if scrape_quarter_rpts is True:
    scrape_bank.grab_bank_rpts()
    reports = ["Assets and Liabilities",
               "Income and Expense"
              ]
    # Extract these reports from the zip files downloaded
    scrape_bank.get_unzipped_file_type(reports)

assets_df = scrape_bank.get_bank_info("Assets and Liabilities")
inc_exp_df = scrape_bank.get_bank_info("Income and Expense")
```

clean_failed_banks is a custom method that goes through the data and does several things:

1) Removes unnecessary columns, primarily those that have lots of missing information.
2) Renames columns into a more human readable, standard python format.
3) Strip several columns of unnecessary whitespace.

4) Convert appropriate columns to numeric/date

engineer_failed_banks is a custom method that adds several ratios and percentages to the data:

1) Debt Ratio = Total Liabilities / Total Assets is a fundamental equation used in accounting to determine the amount of money that a company has borrowed to fund their company.

2) Debt to Equity = Total Liabilities / Total Owner's Equity. For Failed bank data this is especially interesting because most of them have negative Owner's Equity. We calculate positive and negatives separately due to the fact that the results are discontinuous. This is simply due to the fact Owner's Equity can be positive, negative, or zero.

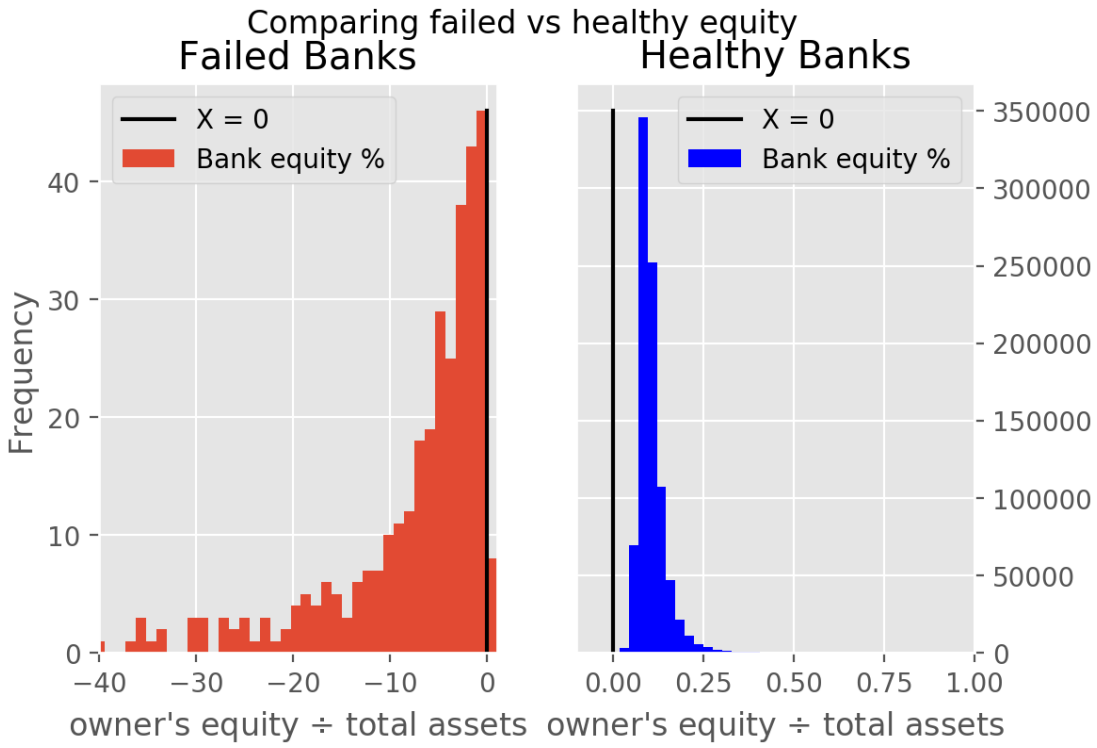
3) Other percentages are calculated for columns that together compose assets and liabilities, such as cash and investments, or administrative liabilities.

Let's do a little bit of engineering to the asset and income/expense reports that we got earlier, and then split them into the *healthy* and *failed* versions

```
In [3]: # Instantiating extra variables "engineering"
        for col in ['total_liabilities', 'owners_equity']:
            failed_df = cne.calculate_ratio(failed_df, col, 'total_assets')

        for col in ['eqtot', 'liab', 'lnatres', 'nclnls']:
            assets_df = cne.calculate_ratio(assets_df, col, 'asset')

        for col in ['eqpp', 'eqcs', 'eqsur', 'equptot', 'eqconsub']:
            assets_df = cne.calculate_ratio(assets_df, col, 'eqtot')

        for col in ['eintexp', 'netinc', 'esal', 'ntlnls', 'eqcdiv']:
            inc_exp_df = cne.calculate_ratio(inc_exp_df, col, 'intinc')

        assets_df['months_before_failure'] = None
        inc_exp_df['months_before_failure'] = None

        # Breaking banks into healthy and failed groups
        failed_indices = assets_df.cert.isin(failed_cert_list)

        failed_assets = assets_df[failed_indices]
        healthy_assets = assets_df[~failed_indices]

        failed_inc_exp = inc_exp_df[failed_indices]
        healthy_inc_exp = inc_exp_df[~failed_indices]
```

All banks are required by law to submit a batch of ~60 reports to the FDIC every quarter. For our first pass, we are choosing to only focus on two of these reports that we believe will be the most useful in finding predictors for bank failures. If needed, our scraper located in scrape_bank is capable of collecting data for any of the 60 reports over any date range given, allowing minute changes to be implemented to expand the project to analyze other reports.

In [4]:
```
fig, axes = plt.subplots(ncols=2)

# Create first plot for failed banks.
failed_df[['owners_equity_pct']].plot(kind="hist", bins=200, ax=axes[0])
axes[0].set_xlim(-40,1)
axes[0].plot([0,0],[0,46], color='black')
axes[0].set_title("Failed Banks")
axes[0].legend(['X = 0', "Bank equity %"])
axes[0].set_xlabel(r"owner's equity $\div$ total assets")

# Create the second plot for healthy banks.
healthy_assets[['eqtot_pct']].plot(kind="hist", color='blue', bins=240, ax=axes[1])
axes[1].plot([0,0],[0,350000], color='black')
axes[1].set_title("Healthy Banks")
axes[1].legend(['X = 0', "Bank equity %"])
axes[1].set_ylabel("")
axes[1].set_xlim(-0.1,1)
axes[1].yaxis.tick_right()

plt.suptitle("Comparing failed vs healthy equity")
plt.xlabel(r"owner's equity $\div$ total assets")
plt.show()
```



The balance sheet for failed banks usually contain negative owner's equity, as seen in the red histogram on the left. Healthy banks, on the other hand, have a positive owner's equity. This suggests that one possible method of detecting a failed bank is to identify those that have decreasing owner's equity. Healthy banks have an average owner's equity of about 11% of their total assets.

Note that failed banks had typically have low assets, incredibly high liabilities, and massively negative owner's equity. We see a much healthier spread in the healthy banks. Their liabilities are still a large portion of the bank's assets, but this matches what a bank should look like because all of the money that customers have in the bank is recorded as a liability. It is surprising how small owner's equity is.

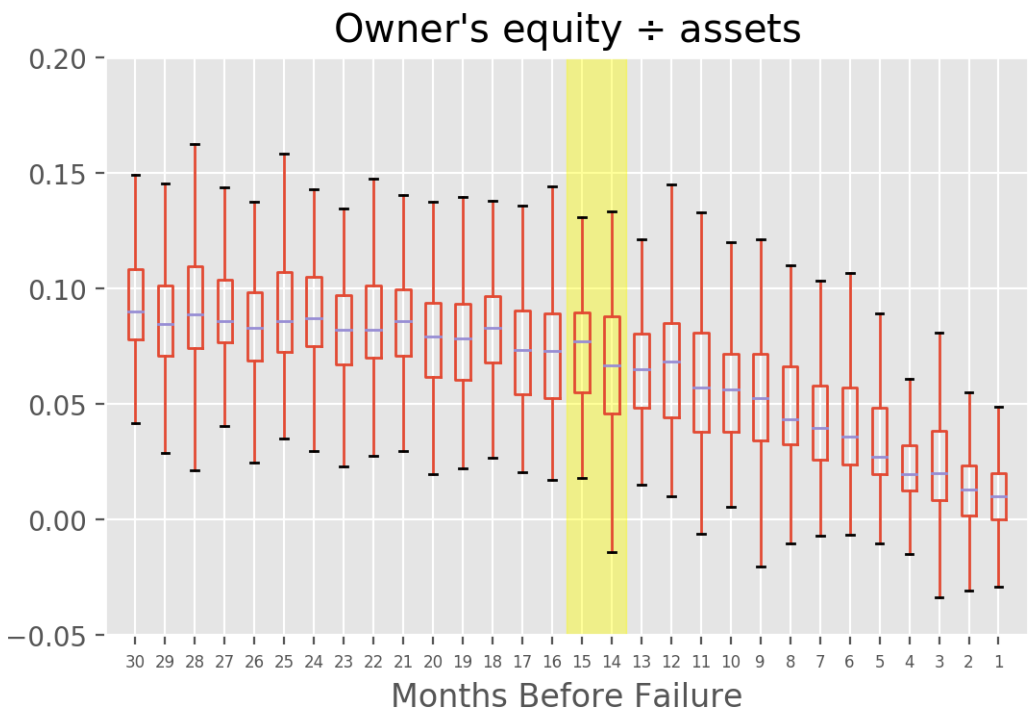
In an effort to identify how failed banks changed up until their point of failure we calculate a timeline for each failed bank and plot the data for a few variables of interest.

In [5]:
```
# Calculate the number of months before a bank fails for each row
months_before_fail = []
date_close = dict()
for i in range(len(failed_assets)):
    row = failed_assets.iloc[i]

    # There are only ~550 certs, we don't need to look them up
    # every time in the dataframe, let's save them for improved performance
    if row.cert not in date_close:
        date_close[row.cert] = failed_summary.loc[row.cert]["Closing Date"]

    date = date_close[row.cert]

    row_year = row.repdte.year
    row_month = row.repdte.month

    fail_year = date.year
    fail_month = date.month

    # This calculates the number of months from this date to date of failure
    months = (fail_year - row_year)*12 + (fail_month - row_month)
    months_before_fail.append(months)

# The data loaded from the FDIC is perfectly uniform
# so the months_before_fail lines up exactly correct for
# income and expenses as well
failed_assets.months_before_failure = months_before_fail
failed_inc_exp.months_before_failure = months_before_fail
```

In [6]:
```
def custom_boxplot(df, title, box_col, xlabel, cols, xlim, ylim, x_size):
    """A custom method made to nicely show boxplots over time for the
    given columns.

    Parameters:
    df (DataFrame) the dataframe which we are plotting.
```

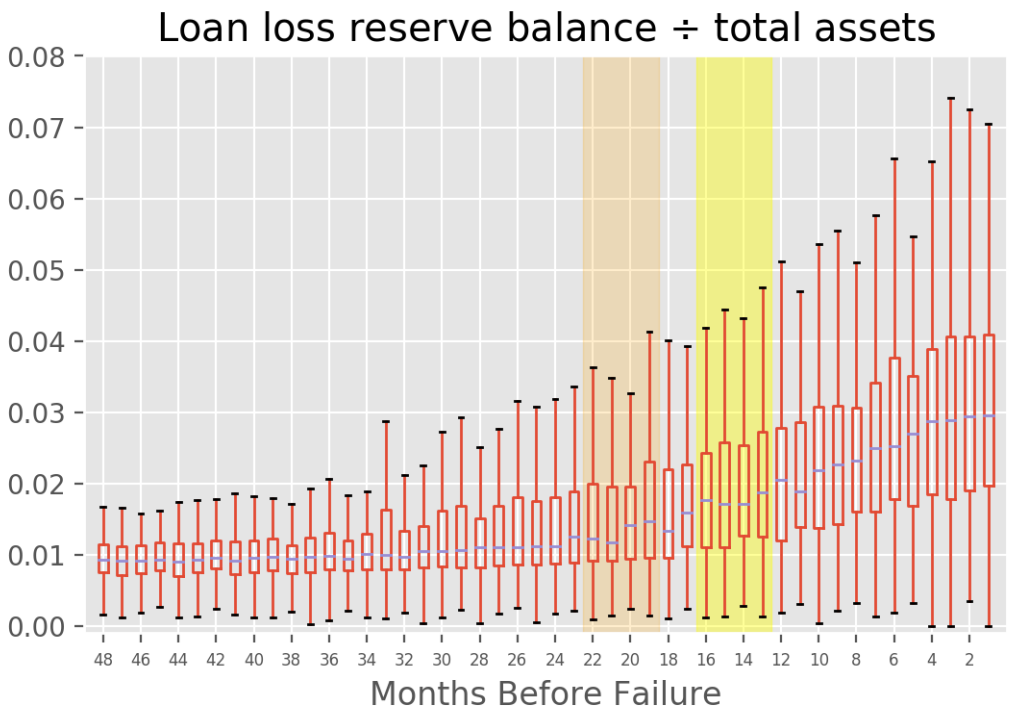
```
        title (string) the title of the plot.
        box_col (string) the column to plot against
        xlabel (string) the label of the x-axis.
        cols (list) a list of columns to plot (must include box_col).
        xlim (list) a list containing the end values of the x-axis.
        ylim (list) a list containing the ned values of the y-axis.
        x_size (int) a number representing the fontsize for x-ticks.

        Returns:
        ax (matplotlib.axes) object for the plot."""
        fig, ax = plt.subplots()
        # Set the outliers to False, they make the plots look messy
        df[cols].boxplot(by=box_col, ax=ax, showfliers=False)
        plt.suptitle("")
        ax.set_title(title)
        ax.set_xlabel(xlabel)
        ax.set_xlim(xlim)
        ax.set_ylim(ylim)
        for item in ax.get_xticklabels():
            item.set_fontsize(x_size)

        # Return the axis in case additions need to be made
        return ax
```

In [7]:
```
data = failed_assets[(failed_assets.months_before_failure <= 30)]
cols = ['eqtot_pct', 'months_before_failure']
title = r"Owner's equity $\div$ assets"
xlim = [31,0]
ylim = [-.05,.2]
ticksize=6
box_col = "months_before_failure"
xlabel = "Months Before Failure"
ax = custom_boxplot(data, title, box_col, xlabel, cols, xlim, ylim, ticksize)
ax.axvspan(15.5, 13.5, color='yellow', alpha=0.4)
plt.show()
```



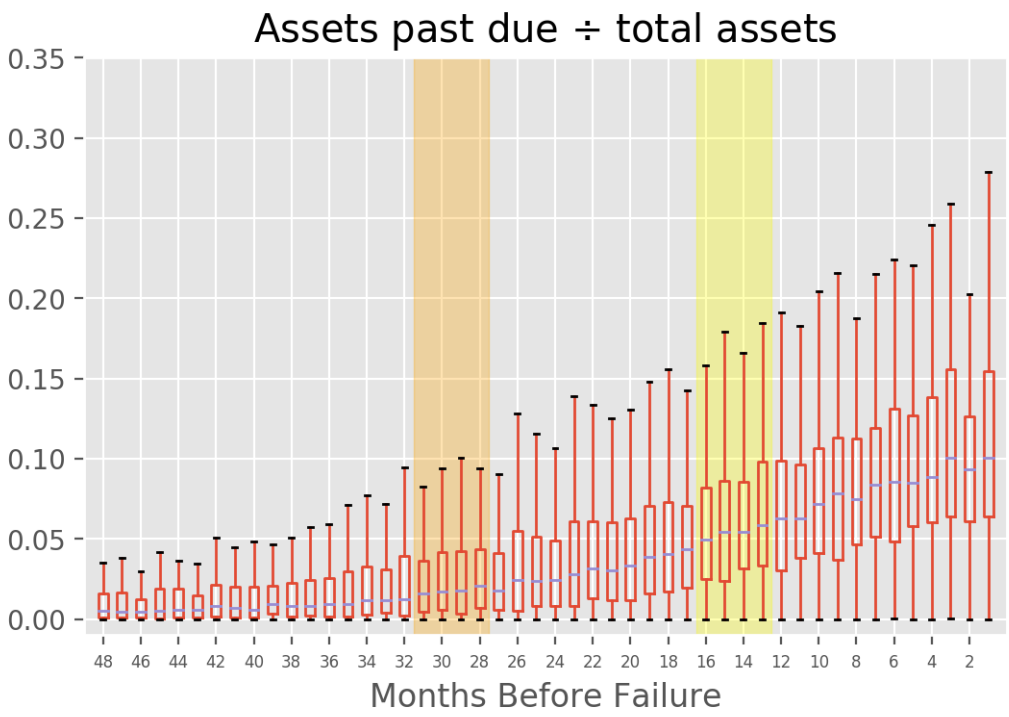
This plot agrees with our initial assumption that the owner's equity continually decreases until the bank fails. There are several outliers where the owner's equity even becomes negative before its failure. The decrease becomes particularly noticeable from about 14 months onward until they fail.

```
In [8]: data = failed_assets[(failed_assets.months_before_failure <= 48)]
        title = r"Loan loss reserve balance $\div$ total assets"
        box_col = "months_before_failure"
        xlabel = "Months Before Failure"
        cols = ["lnatres_pct", "months_before_failure"]
        xlim = [49,0]
        ylim = [-0.001,0.08]
        ticksize = 6
        ax = custom_boxplot(data, title, box_col, xlabel, cols, xlim, ylim, ticksize)
        ax.set_xticks([x for x in range(2,49,2)])
        ax.set_xticklabels([x for x in range(2,49,2)])
        plt.axvspan(16.5, 12.5, color='yellow', alpha=0.4)
        plt.axvspan(22.5, 18.5, color='orange', alpha=0.2)
        plt.show()
```



All banks are required to have a Loan Loss Reserve balance (money that the bank keeps on hand for the loans that they think will default). Interestingly, as the graph above shows, as banks tend toward failure, percent of loan loss reserve to total assets increase. Ironic that as they prepare for loan defaults on loans owed to them, within about 18 months, they will also default on their loans.

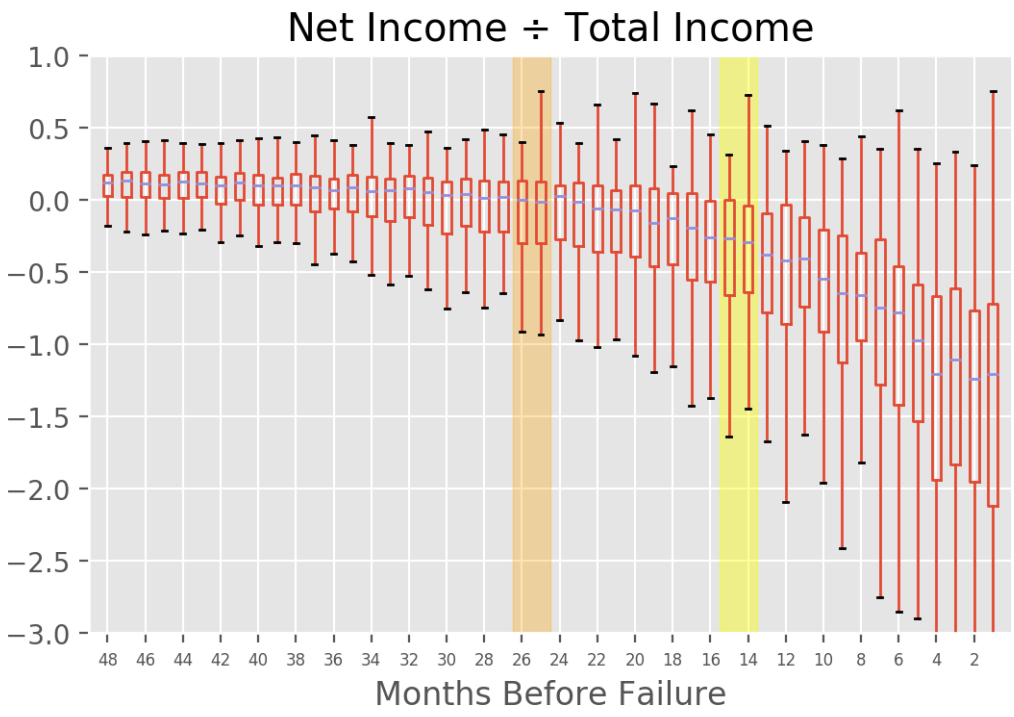
```
In [9]: data = failed_assets[(failed_assets.months_before_failure <= 48)]
        title = r"Assets past due $\div$ total assets"
        box_col = "months_before_failure"
        xlabel = "Months Before Failure"
        cols = ["nclnls_pct", "months_before_failure"]
        xlim = [49,0]
        ylim = [-0.01,0.35]
        ticksize = 6
        ax = custom_boxplot(data, title, box_col, xlabel, cols, xlim, ylim, ticksize)
        ax.set_xticks([x for x in range(2,49,2)])
        ax.set_xticklabels([x for x in range(2,49,2)])
        plt.axvspan(16.5, 12.5, color='yellow', alpha=0.3)
        plt.axvspan(31.5, 27.5, color='orange', alpha=0.3)
        plt.show()
```



In fact, during this same period of time, banks that are headed toward failure begin to have more assets past due. The assets past due are define to be any assets that are at least 90 days past due.

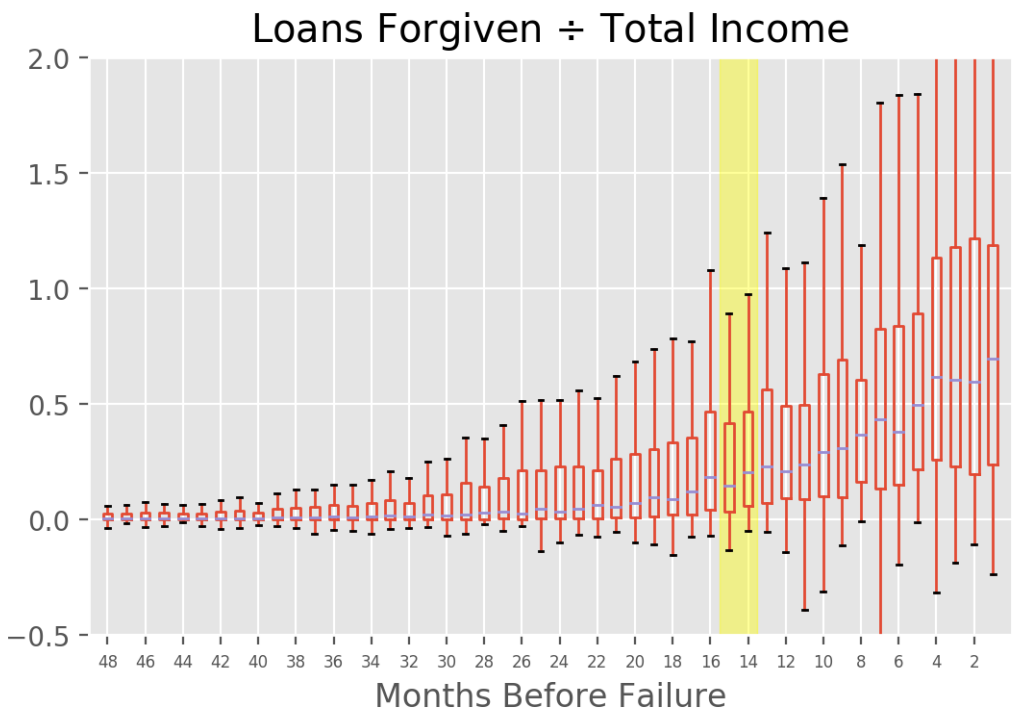
Banks may take out loans for a variety of reasons. A simple example of this would be a bank that takes out a loan to build a new branch, but then, as this bank approaches failure, is unable to keep up on payments of this loan.

```
In [10]: data = failed_inc_exp[(failed_inc_exp.months_before_failure <= 48)]
         cols = ['netinc_pct', 'months_before_failure']
         title = r"Net Income $\div$ Total Income"
         xlim = [49,0]
         ylim = [-3,1]
         ticksize=6
         box_col = "months_before_failure"
         xlabel = "Months Before Failure"
         ax = custom_boxplot(data, title, box_col, xlabel, cols, xlim, ylim, ticksize)
         ax.set_xticks([x for x in range(2,49,2)])
         ax.set_xticklabels([x for x in range(2,49,2)])
         ax.axvspan(15.5, 13.5, color='yellow', alpha=0.4)
         plt.axvspan(26.5, 24.5, color='orange', alpha=0.3)
         plt.show()
```



The orange bar highlights where net income first drops into the negatives, but we can see a sharp decrease over time for net income.

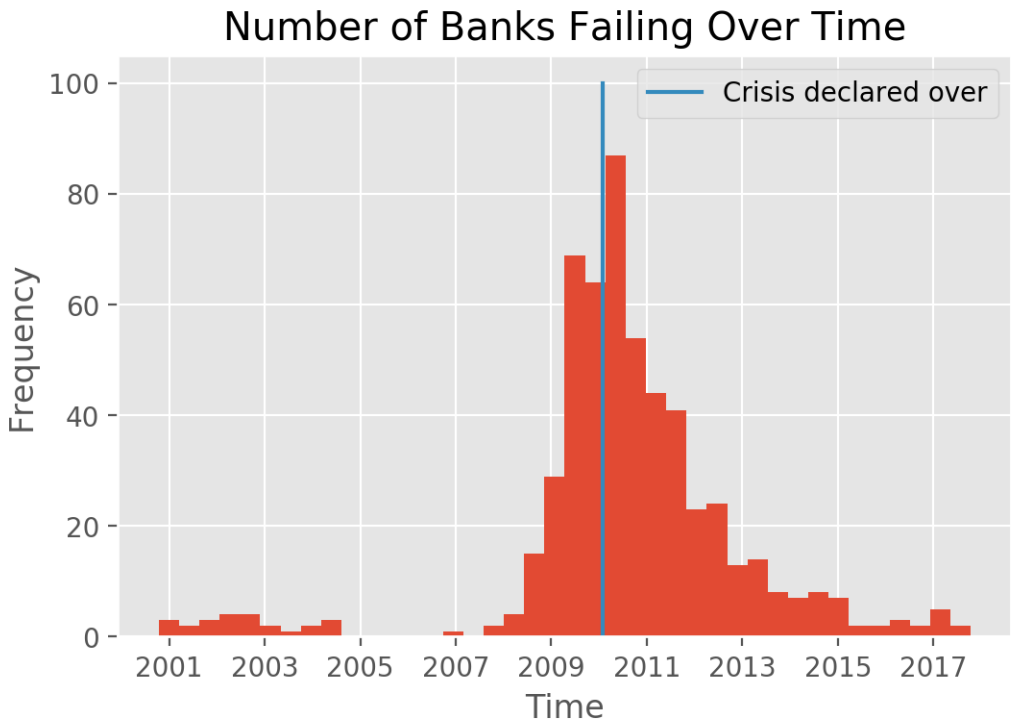
```
In [11]: data = failed_inc_exp[(failed_inc_exp.months_before_failure <= 48)]
         cols = ['ntlnls_pct', 'months_before_failure']
         title = r"Loans Forgiven $\div$ Total Income"
         xlim = [49,0]
         ylim = [-.5,2]
         ticksize=6
         box_col = "months_before_failure"
         xlabel = "Months Before Failure"
         ax = custom_boxplot(data, title, box_col, xlabel, cols, xlim, ylim, ticksize)
         ax.set_xticks([x for x in range(2,49,2)])
         ax.set_xticklabels([x for x in range(2,49,2)])
         ax.axvspan(15.5, 13.5, color='yellow', alpha=0.4)
         plt.show()
```



Here we see that as the banks approach failure, they begin forgiving more loans, recognizing that they will not be able to recover that money.

While examining these failed banks, we can't neglect the impact of current events on overall financial markets. Recall that in 2007-2008 the housing bubble crashed which led to many banks failing. In fact, during the few years that followed this financial crash, more banks were failing than ever before:

```
In [12]: failed_summary['Closing Date'].hist(bins=40)
         plt.title("Number of Banks Failing Over Time")
         crisis_end = datetime(2010,1,27)
         plt.plot([crisis_end,crisis_end],[0,100], label="Crisis declared over")
         plt.legend(loc="best")
         plt.xlabel("Time")
         plt.ylabel("Frequency")
         plt.show()
```



President Barack Obama officially declared an end to the Crisis on January 27, 2010. We'll investigate the banks during this time and after this time separately to see if there are any trends specific to the crisis, about 1/3 of the bank data we have is from the crisis.

Through this analysis, it is clear that about 15 months before failure *banks act differently*. They forgive more loans, increase their loan loss reserve, and delay or stop payment on their loans they owe. Though we have clearly shown coorelation between failure and these practices, we are left to wonder whether banks do these things as they realize that they are failing, whether they fail in part because of these practices, or whether there are other variables that cause both these practices and the bank to fail.

In [ ]: