# Prediction Of Company Bankruptcy

Advance hybrid analysis on the business regulations of the Taiwan Stock Exchange Company bankruptcy dataset. Developed a machine learning model based on multiple algorithms and evaluators to identify and predict a similar type of company status.



## Key Technologies :

- PCA
- Decision Tress
- Random Forests
- AdaBoost
- Gradient Boost
- Logistic Regression
- KNN
- XGB Boost
- Multi-layer Perceptron
- Cross-validation


Project By:

**AMLAN NAG**


COMP 4980, Special Topics: Machine Learning

Date : 14 April 2022

## Table of Contents

## Introduction

In this era, the surge of startups or small-scale companies is enormous. Where a small business or new business mostly does not have sufficient capital to scale up their company. Same time with limited resources, they have to spend money wisely to balance out the company's natural growth. In terms of wise decisions, it should not come from human instinct or a dream; the business must need to drive through data, which is also called a Data-driven business. With the help of machine learning, one can predict their company status based on multiple economic factors. To solve this issue with the help of machine learning, numerous authors have developed a machine learning model, and there are various periodical research papers conducted on this topic with several types of analysis. We need to keep in mind that inaccuracy in bankruptcy predictions can have a severe financial impact and result in a disastrous blow to business owners, partners, the community, and the entire economy. As a result, bankruptcy prediction is of interest to the company's internal management, the audit, and public authorities since it influences their decision-making. As a result, strengthening the capacity to foresee bankruptcy appears to be particularly crucial.

Over the span of decades, there were multiple bankruptcy models developed. Ottman developed a multivariate statistical technique to categorize the firm based on financial report data [2]. Ohlson (1980) applied logistic regression analysis to this topic [3]. Though multiple authors have implemented different techniques with it and developed different models as time changes, those results are not accurate and effective enough to support newer businesses. In order to avoid similar devastating disasters in the future, more robust predictive models are required. The financial market is affected on several fronts by company and enterprise bankruptcy. Thus, anticipating insolvency among organizations by monitoring various characteristics becomes even

more critical. More excellent knowledge of bankruptcy and the capacity to foresee it will influence lending institution profitability throughout the world.

In this project, I would try to develop a hybrid analysis model which will cover various techniques and multiple ML algorithms. It would integrate the data preprocessing step with the algorithm, which would significantly increase classification performance. Basically, it increases or decreases the size of the training data space by adding or eliminating samples to lessen the discriminatory behavior of imbalanced data.

In this project, I am using the company bankruptcy dataset from Taiwan Economic Journal for the years 1999 to 2009 based on the business regulations of the Taiwan Stock Exchange, which is publicly available on kaggle.com. Using this dataset, I would perform data modelling, cleaning and scaling, then initial exploration with Principal Component analysis and using a decision tree would observe how the dataset is behaving and performing with algorithms. After that, I would develop a machine learning hypothesis using several algorithms like Random Forests, Adaboost, GradientBoost, Regression, SVM, KNN and XGBoost. After that, I would implement Neural Network and perform cross-validation to justify the experimental result. The machine learning module would include parameter tunning and Grid Search, a standard machine learning procedure. For the result, the comparison would use classification matric, precision and cross-validate with all algorithm results to ensure that I implement the most accurate algorithm for prediction. This project would establish a hybrid approach of analysis that focuses on algorithm accuracy or performance and focuses on tunning and developing an accurate machine learning model.

## Data Collection and Description

I collected the dataset from Keggel.com in CSV format which is publicly available for any sort of analysis and research.

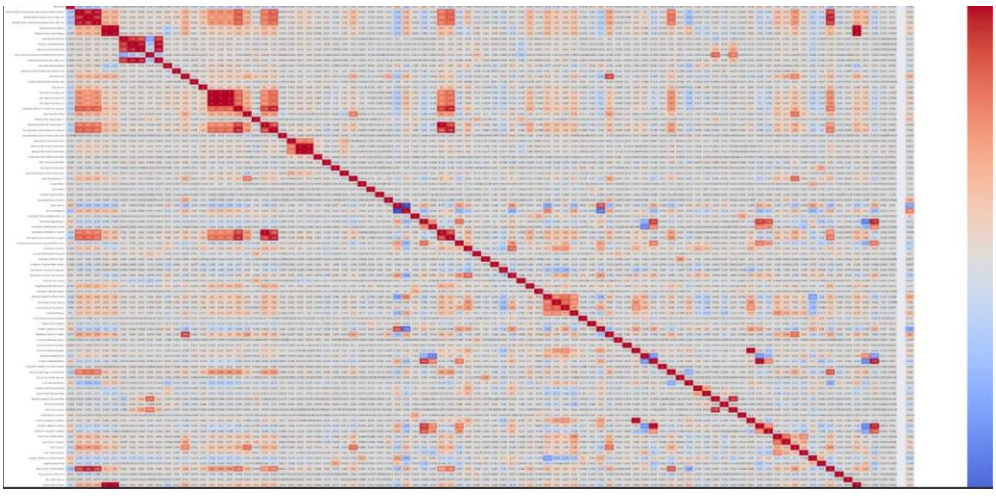
The dataset is about "Bankruptcy data from the Taiwan Economic Journal for 1999–2009". The Taiwan Stock Exchange's corporate laws were used to define company bankruptcy. This dataset has been standardized for similar case analysis and prediction on a larger scale. The dataset is fundamentally scaler data with a binary Class, and it contains 96 columns and 6819 rows. The features are mainly financial factors or standards that can determine a company's credit status and evaluate factors such as Net Income Flag, Equity to Liability, Etc. The data type is categorized into integer and float values. This is a scalable and standardized dataset to implement machine learning to predict a company's status, whether it will go bankrupt or survive. The dataset contains a variety of finical factors, which could help all new startup companies or entrepreneurs to do the assessment with their business or determine the best path for operating their business.

For ease of use- I uploaded the CSV file to a GitHub repository to call the dataset from the cloud platform. In order to develop the machine learning model and analysis purpose, I chose python as our development language because python has enriched libraries and developed functions for machine learning. In terms of IDE, I created a prediction program file in Google Colab (Jupiter Lab). I named the program Hybrid Analysis of bankruptcy and then called all required packages that I would need to perform the analysis. However, there might be multiple packages or libraries that I would call in the program as new requirements. After that, I called the dataset from the GitHub repository using the pandas' framework.

## Data Analysis

In the data analysis part, I checked the dataset with data.head command and using display. precision explore the data tail to see the values in two decimal places. After that, I run a command of data info which provides the dataset attributes details with the data type. Data information provides that the dataset contains float and integer types of data. After that, I run data. Describe functions that provide the total count of the dataset and then the mean value for each feature in the dataset. The procedure allows provided information on standard deviation, min value and others which helped a lot to understand the dataset.

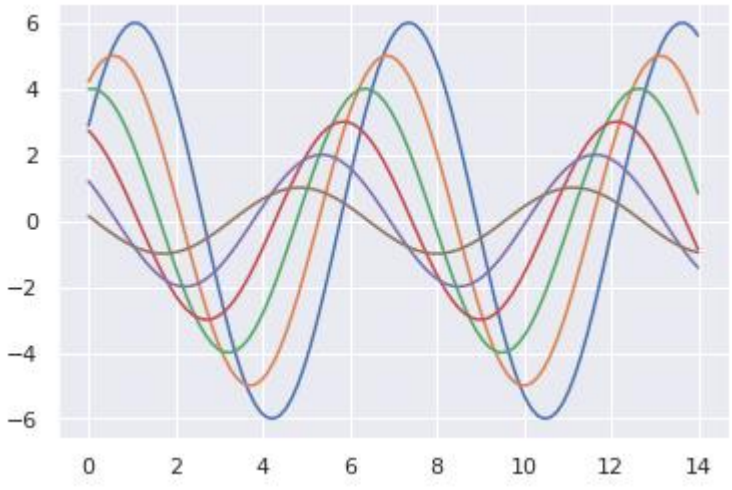
After that, to understand the importance of the features towards the dataset and their attributes correlations, run the data.corr with heatmap to visualize the dataset with a map where it aligns with linear line and connect and show the values of attributes relation in range of positive 1 to negative 1. The correlation efficiency only measures linear correlation. Then sort the values of correlation and rank based on a false parameter which gives the result of the value of the topmost features to 1.



From there, I could see the essential features that would assist the machine learning model in predicting the attributes' features value towards the label and decided to help for different input

to indicate that it is a sample for bankrupt or still surviving. After that, I run a function to print the histogram of the dataset, which represents the distribution of data. This function data.hist(), on each series in the data frame, resulting in one histogram per column. From  I received a similar sort of result. After the process, I was able to know which features are most meaningful to the ongoing machine development.

In order to develop an accurate hybrid analysis model and get the best accuracy with a less false negative value, I have performed data modelling and preprocessing of the dataset. First of all, run isnull.sum function that would fill up all null values in the dataset. After that, count the total number of bankruptcy and survival companies and visualize their ratio with a bar graph. After that, to observe more in detail, use a simple function to plot some offset sine waves; this will allow us to view the various style parameters that I can adjust.

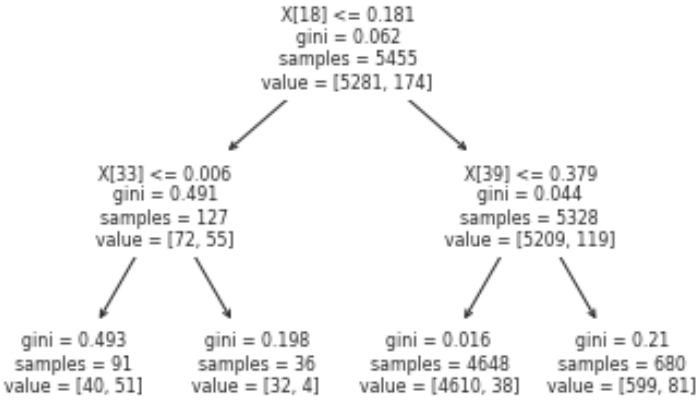



Then using sinplot() vizulize the data. Observing that, I performed one hot encoding with pd get dummies this particular command. One-hot encoding turns categorical data into a binary vector representation and unique value in a column. Depending on whether the value matches the column heading, the values in this column are represented as 1 or 0. The Pandas get dummies() function returns a data frame containing dummy variables for the column provided. After that, I displayed all values of the column after encoding.

Then using le fit, perform prepossessing to align with the class type. LabelEncoder encodes labels with a value between 0 and 1 of the Bankrupt class. After completing all the mentioned experiments, Now it is time to explore the data with different analyzing techniques.

## Data Exploration

In the exploration stage, the first assigned 33 attributes to a feature object. These 33 attributes are the most meaningful attributes that I have received from the previous experiments. To retrieve data and assign it to an x-axis data using iloc, I fit these features to x and the class Bankrupt to the y axis. Then, I called the standard scaler function and transformed and fit the data. Using the standard scaler helped to scale the data to the dataset and make it function to use ml techniques. Then called the decomposition package to use PCA. In PCA function, used to 2 value as n_componets and fit transform to principal components. And I have completed data framing for the principal Component and divided the dataset into 2 sections. Then concat the values with the bankrupt class. Then print the head for final principal df, and it shows the result. And then, using a variance ratio see the array value of 0.37 and 0.11. This describes only 37% of the dataset variance lies along with the first PC, and 11% lies along with the second PC; this leaves 52%, which is reasonable to assume that the third Pc probably caries most information. After that, I split the dataset into a train and test model where the test size is 20%. Then, the Decision Tree Classifier was utilized, using the standard object clf and assigned to the classifier and set max_depth to 2. Then plotted the tree and where the model split to 5281 and 174 values, and the Gini was 0.062 and observed the continuous tree splitting. These two experiments provide the rational variance of the dataset and Gini works on categorical variables, provides outcomes, either be successful or failure. and hence conducts splitting. The Gini value usually varies from 0 to 1, where is the first distribution; the Gini was denoted with 0.06, and then when it classified and split into two trees




the Gini weight developed to 0.49, and other 0.04 then continues splitting it improved, and the information relates been established and shown.

## Experimental Method

In the experimental phase, I first copy the data with an attribute to execute different tests and algorithms. As my, the data is mainly classified into float and int values, and the int value refers to the class file of the dataset, so I dropped the class file from the x-axis and assigned it to the y-axis. After that, with from help of scikit-learn packages, split the data into a testing and training model. Where the test size was set to 20%. Then checked the distribution ratio. After enabling sklearn, I called the random forest classifier and set up the n_estimators o 10 to implement the algorithm towards the scaled dataset. Calling classifier predicts function and assigned to attribute after that with, standard practice assigned the y-test value and the predicted value to an accuracy function and print the accuracy. Then with the help of the classification report and confusion matrix, print the report for the particular algorithm. In the accuracy function, I called this algorithm with acc1 because, in the end, I would like to develop a comparative graph and table that represent the entire testing results. After observing with a decision tree, I thought Random

Forest would make more sense to use with the data as because it would enable entropy of data and splitting tree it makes aligns with the class file.

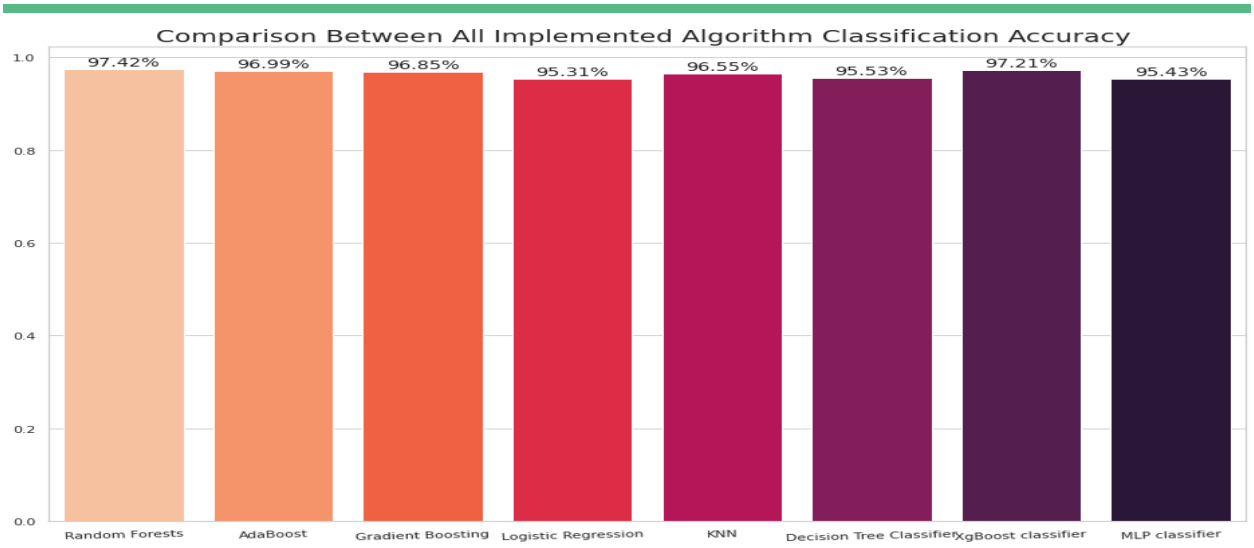
As some multiple algorithms and classifiers are pre-modelled and have callable libraries to be implemented with python. Then I called AdaBoost Classifier to observe the machine learning model accuracy, precision and etc. using the algorithm. Where this boosting algorithm, with its low-classifying features, would provide the highest accuracy for the model. The classifier implemented the SAMME algorithm, which is a stage-wise additive modelling. SAMME.R algorithm presents all the models with an equal weight of one. It outputs a real value. The accuracy of this algorithm was about 96% which is significantly good. Same as before, implemented with acc2 and assigned the accuracy classification function and confusion matrix to display the insights. Later, pandas import errors and sklearn the mean square error because the next algorithm I was planning to implement was Gradient Boosting Classifier which requires assigning n_estimator to an error constant for the mean square error value. After implementing the algorithm, I printed the accuracy and confusion matrix where the accuracy was assigned to acc3. Now, following the same strategy called the required packages for Logistic Regression, was setting up the random state to zero aligned the classifier using the fit function to train the model. Then assigned, the classifier predicts to the y_pred attribute then assigned it acc4 and prints the classification and confusion matrix. Then printed it with acc5 and got the result of 96%, which is really significant. After that, I implemented the KNeighbors algorithm and set up k-neighbour to 5 and then the fitted model accuracy was 96%. Then I implemented the decision tree classifier algorithm and its entropy and got results of 95%. At this stage, I called the XGB Classifier and assigned it with a classifier and fir the model to train a model. And then printed the classifier accuracy, and the result was 97% which is the highest so far.

At this stage, I prepared the program for implementing Multilayer Perceptron neural network. I called all required packages that I will require to develop the model. I again copied the data to another attribute. My focus was to copy to a new data frame so that the previously trained model does not get biased. I could explore the data scathingly with a different method. Then again same assigned the data to x and the class "Bankrupt" to the y-axis. After that split, the dataset was not assigned the ratio of the train and split model. After that, I fitted the model to test, valid and train the model. After that, I implemented the Keras model sequence with model attribute and used relu and compile model with mean square and developed the model to history function and alleged with the train and test split model. Using the mse test evaluate the model and fit the model to y_pred and print the epochs. After that, print the coefs to the relation of data in the arrays of the MLP process. Then printed is the hidden layer, and it appears to be 10, 5 ratios. After that, I implemented the MLP classifier with clf and where set the hidden layer that I have received and max iter to 1000 and implemented the relu function. And then implemented the model to prediction function and assigned to accuracy function and print the value with acc8 and print classification result and confusion matrix. And from MLP classification result was 95%.

## Result and Analysis

To summarize the project result, the best result of this project was the accuracy of 97% with the XGB Classifier. Where the recall for '0' was 1 and for '1' was 0.28. The average f1 score was 97%, and the average precision was 85%, and the weighted avg was 97%. The confusion matrix was (1313, 5) and (33,13). Compared to the standard machine learning model, this accuracy is significant. Multiple processes have been implemented to acquire and develop a model that understands the data and predicts the class based on the model. In the experimental section of the project, I have printed all utilized algorithms and evaluation metrics using a classification matrix

and confusion matrix with accuracy. The results of the machine learning model have a significant improvement in terms of before scaling up and now. Cross-validation is the proof of the improvement; after that, complete experiment where all results are described in detail. In the result and analysis sections, I first assigned the y-pred attribute to the classifier to predict and get results for accuracy. Where for the cross-validation, I called the cross-val model from the sklearn model selection and then assigned the estimator to the classifier and X to x_train and y to y_train and cross-validation(cv) to 10. For the above-implemented algorithm, the Accuracy validation was 96.88%, and the Standard Deviation Ratio was 1.04. After that, I developed a comparative table calling all accuracy results with their associate algorithm to see all results at a glance and observe the performance. Using a fancy grid table, printed all accuracy. After that, using the list method combined and append all accuracy to a list object. My idea is to develop a comparative bar graph for the result; I was inspired by this step from another research on this topic, which appears very effective in finding the best algorithm quickly with accuracy. Inspiring from that, I assigned all the algorithm names to graphlist_1 and the accuracy result to the graphlist. Then using the sns barplot, I allocated the x-axis to graph_list1 and y to graphlist and color palette. A function that I refactored for adjusting the bar graph visualization for height and weight adjustment based on the result. After the program visualized a bar graph with all algorithm results, and as we can see from it, the XGBoost has the most accurate result. That's the end of the project result and now let’s analyze the results.




Let's discuss about the accuracy result, what does actually mean, and what the ml model is performing with the algorithm. The accuracy is the ml model that accurately classifies the data here; the models are basically aligned and compare the data with each entire of attributes and then develop an aligned relationship with the class of Bankrupt based on the boolean type 0 or 1 and then predict the data to its class. So mainly, it compares all features values and assigned with the class and continues compassion after that the process It was able to predict the class based on the data. In this case, the accuracies also represent how the model reads the data and fits into the testing model.

To conclude the project, this particular project helped to develop a model that could easily fit into a predictive model and be deployed to an AI-based predictive agent. Also, comparing all previous research on this topic with a machine learning algorithm, the accuracy of 97.21% is one of the highest results that the research fraternity could achieve. As I described in the introduction and my project goal was to develop a model that could help to predict the company's status and the model could be assigned with a predictive function and take user input for all features, and then the model is able to predict either the company is bankrupt or not. The predictive would be

successful with a high accuracy rate, and by these experiments, the result satisfied the project objectives.

## Resources

**Dataset:** https://www.kaggle.com/datasets/fedesoriano/company-bankruptcy-prediction

**Project Notebook:**

https://colab.research.google.com/drive/1xDq78hHImZcGzaLat89b4VFIKdtSwQS7?usp=sharing

**Data Storage Link:** https://github.com/amlannag6/ML_Testings-Datasets/blob/main/Bankruptcy%20Prediction.csv

## References


*Classification Example with XGBClassifier in Python*. (7/04/2019 ). From Data TechNotes : https://www.datatechnotes.com/2019/07/classification-example-with.html

*Getting started with the Keras Sequential model*. (n.d.). From Keras 2.0.2 Documentation: https://faroit.com/keras-docs/2.0.2/getting-started/sequential-model-guide/

John Hunter, D. D. (Copyright 2002 - 2012). *matplotlib.pyplot.hist*. From matplotlib: https://matplotlib.org/3.5.0/api/_as_gen/matplotlib.pyplot.hist.html

License), s.-l. d. ( 2007 - 2022). *sklearn.ensemble.AdaBoostClassifier*. From scikit-learn: https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.AdaBoostClassifier.html

License), s.-l. d. ( 2007 - 2022). *sklearn.ensemble.GradientBoostingClassifier*. From scikit-learn: https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.GradientBoostingClassifier.html

License), s.-l. d. ( 2007 - 2022). *sklearn.neural_network.MLPClassifier*. From scikit-learn: https://scikit-learn.org/stable/modules/generated/sklearn.neural_network.MLPClassifier.html

License), s.-l. d. (2007 - 2022). *6.3. Preprocessing data*. From scikit-learn: https://scikit-learn.org/stable/modules/preprocessing.html

License), s.-l. d. (2007 - 2022). *sklearn.decomposition.PCA*. From scikit-learn: https://scikit-learn.org/stable/modules/generated/sklearn.decomposition.PCA.html

License), s.-l. d. (2007 - 2022). *sklearn.ensemble.RandomForestClassifier*. From scikit-learn: https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html

License), s.-l. d. (2007 - 2022). *sklearn.linear_model.LogisticRegression*. From scikit-learn: https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LogisticRegression.html

License), s.-l. d. (2007 - 2022). *sklearn.metrics.accuracy_score*. From scikit-learn: https://scikit-learn.org/stable/modules/generated/sklearn.metrics.accuracy_score.html

License), s.-l. d. (2007 - 2022). *sklearn.model_selection.cross_val_score*. From cikit-learn: https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.cross_val_score.html

License), s.-l. d. (2007 - 2022). *sklearn.model_selection.train_test_split*. From scikit-learn: https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.train_test_split.html

License), s.-l. d. (2007 - 2022). *sklearn.preprocessing.StandardScaler*. From scikit-learn: https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.StandardScaler.html

License), s.-l. d. (2007 - 2022). *sklearn.tree.DecisionTreeClassifier*. From scikit-learn: https://scikit-learn.org/stable/modules/generated/sklearn.tree.DecisionTreeClassifier.html

License), s.-l. d. (2010 - 2014). *sklearn.neighbors.KNeighborsClassifier*. From scikit-learn: https://scikit-learn.org/0.15/modules/generated/sklearn.neighbors.KNeighborsClassifier.html

Lukic, M. (2013-2022). *One-Hot Encoding in Python with Pandas and Scikit-Learn*. From StackAbuse: https://stackabuse.com/one-hot-encoding-in-python-with-pandas-and-scikit-learn/

MONDAL, S. (2022, February). *Company Bankruptcy Prediction, ACC: 97%.* From Kaggle: https://www.kaggle.com/code/sanjoymondal0/company-bankruptcy-prediction-acc-97#Split-the-data-set-(for-PCA)

Richard E. Neapolitan, X. J. (2007). *Bankruptcy Prediction*. From sciencedirect: https://www.sciencedirect.com/topics/computer-science/bankruptcy-prediction

*seaborn.heatmap*. (Copyright 2012-2021). From Seaborn : https://seaborn.pydata.org/generated/seaborn.heatmap.html

Wang H, L. X. (2021). *Undersampling bankruptcy prediction: Taiwan bankruptcy data*. From journals.plos: https://journals.plos.org/plosone/article?id=10.1371/journal.pone.0254030

Waskom, M. (Copyright 2012-2021). *Controlling figure aesthetics*. From seaborn: http://seaborn.pydata.org/tutorial/aesthetics.html