BANA 620 PROJECT REPORT: DESCRIPTIVE AND PREDICTIVE ANALYTICS APPLIED TO THE SKILLED NURSING FACILITY COST REPORTS

Joshua Cabal

California State University, Northridge

May 2, 2024

Contents

1 Executive Summary

1.1 Abstract

This study aimed to identify the critical operational and financial metrics that predict the financial performance, specifically net income, of skilled nursing home facilities. By analyzing various factors, this project aims to determine the features that are most important to predict net income and to determine the feasibility behind investing in skilled nursing home facilities.

1.2 Key Findings

The research revealed that the most significant financial predictors of a skilled nursing facility's success, which was determined to be net income, include net profit margin, net income from patients, inpatient revenue, and salary costs per bed. In addition to these financial metrics, several non-financial indicators such as deficiency scores from Cycle 1, Cycle 2, and Cycle 3 were also vital. Although the geographical location of the facility did not generally present as an important factor relative to the other dataset features, the analysis did find that facilities in states such as New York, Connecticut, California, and Virginia exhibited superior performance.

1.3 Recommendations

Given these findings, it is recommended that investments in skilled nursing facilities be contingent upon the engagement of experienced personnel to manage such projects. This oversight will ensure optimal staffing and operation within markets demonstrating a clear demand and historical success for these services.

2 Introduction

Due to the significant number of baby boomers exploring nursing home options, this project takes the perspective of a company which has been tasked with determining whether it is advisable to invest in nursing homes.

2.1 Context and Background Information

To facilitate this analysis, utilize U.S. nursing homes datasets spanning the years 2015 to 2021. These datasets will serve as the foundation for the investigation, enabling the examination of trends, performance indicators, and potential investment opportunities within the nursing home sector.

2.2 Objectives

This study is guided by a series of specific objectives designed to deepen the understanding of the financial dynamics within the U.S. nursing home sector over the selected period. These objectives are outlined as follows:

1. **Evaluate** the overall financial performance of nursing homes in the United States during this period.
2. **Identify** the influential factors that affect the financial performance of these facilities.
3. **Determine** the most significant factors that impact the performance of nursing homes, with a special emphasis on distinguishing between operational and external influences.
4. **Describe** the trends observed in the performance of nursing homes and the dynamics of these influential factors over time.
5. **Analyze** the specific impact of the COVID-19 pandemic on the financial and operational stability of nursing homes.

In support of these objectives, the project will incorporate a comprehensive exploratory data analysis (EDA) to extract key insights from historical data spanning several years. Additionally, predictive analytics techniques will be employed to ascertain the most critical features influencing the sector's performance. This analysis aims to provide actionable insights and data-driven recommendations to stakeholders considering investments in this industry.

3 Methodology

The dataset was procured from publicly available sources and all software utilized is open source. Many of the methods used are regression models or nearest neighbors approximatioons, and by utilizing the source code of the repository, all findings will be easily replicable.

3.1 Software and Tools

The analysis was conducted using Python (version 3.11.4) within the **Jupyter Notebooks** environment. Several Python libraries for their capabilities in data analysis and machine learning.

1. **Pandas**: Data structures and operations for manipulating numerical tables and time series.
2. **NumPy**: Support for large, multi-dimensional arrays and matrices, along with a large collection of high-level mathematical functions to operate on these arrays.
3. **Scikit-learn**: A free and open-source machine learning library.
4. **Matplotlib** and **Seaborn**: For creating static, animated, and interactive visualizations.

These tools were chosen due to their robust functionality and broad acceptance within the data science community, which together facilitate efficient and effective data processing, analysis, and model development.

3.2 Preprocessing Scalers

Two different numerical scalers were used and compared. Below is the description of each method.

1. **Standard Scaler:** Standardize features by removing the mean and scaling to unit variance. The standard score of a sample x is calculated as:

   $$z = \frac{x - u}{s}$$

   Centering and scaling happen independently on each feature by computing the relevant statistics on the samples in the training set. Mean and standard deviation are then stored to be used on later data using transform.

   Standardization of a dataset is a common requirement for many machine learning estimators: they might behave badly if the individual features do not more or less look like standard normally distributed data (e.g. Gaussian with 0 mean and unit variance).

   For instance many elements used in the objective function of a learning algorithm (such as the RBF kernel of Support Vector Machines or the L1 and L2 regularizers of linear models) assume that all features are centered around 0 and have variance in the same order. If a feature has a variance that is orders of magnitude larger than others, it might dominate the objective function and make the estimator unable to learn from other features correctly as expected.

2. **Robust Scaler:** Scales features using statistics that are robust to outliers.

   This Scaler removes the median and scales the data according to the quantile range (defaults to IQR: Interquartile Range). The IQR is the range between the 1st quartile (25th quantile) and the 3rd quartile (75th quantile).

   Centering and scaling happen independently on each feature by computing the relevant statistics on the samples in the training set. Median and interquartile range are then stored to be used on later data using the transform method.

   Standardization of a dataset is a common preprocessing for many machine learning estimators. Typically this is done by removing the mean and scaling to unit variance. However, outliers can often influence the sample mean / variance in a negative way. In such cases, using the median and the interquartile range often give better results.

3.3 Machine Learning Models

Several different models were used and compared in this study to determine model suitability. Below are the details related to each method.

1. **Linear Regression**: Ordinary least squares Linear Regression. LinearRegression fits a linear model with coefficients w = (w1, ..., wp) to minimize the residual sum of squares between the observed targets in the dataset, and the targets predicted by the linear approximation.

   $$\text{minimizes: } \frac{1}{2n}\sum_{i=1}^{n}\left(y_i - (\beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \cdots + \beta_p x_{ip})\right)^2$$

   where:

   - n is the number of observations in the dataset.
   - y_i represents the observed value of the dependent variable for the i-th observation.
   - x_{ij} is the value of the j-th explanatory variable for the i-th observation.
   - β_j denotes the coefficient for the j-th explanatory variable, quantifying the effect of this variable on the response.
   - β_0 is the intercept, representing the expected mean value of y_i when all explanatory variables are equal to zero.

2. **Lasso Regression**: Linear Model trained with L1 prior as regularizer (aka the Lasso). Regularization improves the conditioning of the problem and reduces the variance of the estimates. Larger values specify stronger regularization.

   $$\text{minimizes: } \left\{\frac{1}{2n}\sum_{i=1}^{n}(y_i - \mathbf{x}_i^\top \beta)^2 + \alpha\|\beta\|_1\right\}$$

   where:

   - n represents the number of observations in the dataset.
   - y_i is the observed value of the dependent variable for the i-th observation.
   - $\mathbf{x}_i$ is the vector of explanatory variables (features) for the i-th observation.
   - $\mathbf{w}$ denotes the vector of coefficients (including the intercept and slopes) of the linear model.
   - α is a non-negative regularization parameter that controls the strength of the L1 penalty, encouraging sparsity in the vector of coefficients.
   - $\|\mathbf{w}\|_1$ represents the L1 norm of the coefficients, which is the sum of the absolute values of the coefficients.

3. **Ridge Regression**: Linear least squares with L2 regularization. This model solves a regression model where the loss function is the linear least squares function and regularization is given by the l2-norm

   $$\text{minimizes: } \frac{1}{2n}\sum_{i=1}^{n}\left(y_i - (\beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \cdots + \beta_p x_{ip})\right)^2 + \lambda\sum_{j=1}^{p}\beta_j^2$$

   where:

   - n is the number of observations in the dataset.
   - y_i represents the observed value of the dependent variable for the i-th observation.
   - x_{ij} is the value of the j-th explanatory variable for the i-th observation.
   - β_j denotes the coefficient for the j-th explanatory variable, quantifying the effect of this variable on the response.
   - β_0 is the intercept, representing the expected mean value of y_i when all explanatory variables are equal to zero.
   - λ is a non-negative regularization parameter that controls the strength of the penalty applied to the size of the coefficients.

4. **Elastic Net**: Linear regression with combined L1 and L2 priors as regularizer.

$$\text{minimizes: } \frac{1}{2n}\sum_{i=1}^{n}(y_i - (\beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \cdots + \beta_p x_{ip}))^2 + \lambda\left(\alpha\sum_{j=1}^{p}|\beta_j| + \frac{1-\alpha}{2}\sum_{j=1}^{p}\beta_j^2\right)$$

   where:

   - n is the number of observations in the dataset.
   - y_i represents the observed value of the dependent variable for the i-th observation.
   - x_{ij} is the value of the j-th explanatory variable for the i-th observation.
   - β_j denotes the coefficient for the j-th explanatory variable, quantifying the effect of this variable on the response.
   - β_0 is the intercept, representing the expected mean value of y_i when all explanatory variables are equal to zero.
   - λ is a non-negative regularization parameter that controls the overall strength of the penalty.
   - α is a parameter between 0 and 1 that balances the contribution of L1 and L2 penalties: α for the L1 component and $(1-\alpha)$ for the L2 component.

5. **K Neighbors Regressor**: Regression based on k-nearest neighbors. The target is predicted by local interpolation of the targets associated of the nearest neighbors in the training set.

$$\text{prediction for } x_q = \frac{1}{K}\sum_{i\in N_K(x_q)} y_i$$

   - x_q is the query point for which the prediction is to be made.
   - K is the number of nearest neighbors considered for making the prediction.
   - $N_K(x_q)$ is the set of indices of the K nearest neighbors to x_q.
   - y_i are the observed values of the dependent variable for the neighbors.

4 Data Description

Below is an overview of the data sets used, including sources, size, and characteristics of the data. Highlight any data cleaning or preprocessing steps undertaken to prepare the data for analysis.

4.1 Skilled Nusing Facility Cost Report

Medicare-certified institutional providers are required to submit annual cost reports. These data files contain the highest level of cost report status for cost reports in each reported fiscal years. The cost report contains provider information such as facility characteristics, utilization data, cost and charges by cost center (in total and for Medicare), Medicare settlement data, and financial statement data. CMS maintains the cost report data in the Healthcare Provider Cost Reporting Information System (HCRIS). Skilled Nursing Facilities (SNF) submit their cost report data to HCRIS using form CMS-2540-2010. The reports used are from the years 2015 through 2021.

4.1.1 Data Loading and Handling Missing Values

Each file comprises annual cost reports for over 15,000 skilled nursing facilities, all stored in separate CSV files for each of the years under study. To facilitate analysis, these files required concatenation into a single DataFrame. Initially, inconsistencies in column headers across the files for the years 2020 and 2021 were modified to match with those of previous years. This matching was manually executed in Microsoft Excel by renaming similar columns accordingly and discarding unmatched columns. Additionally, a `Year` attribute was appended to each record during the import step to preserve the, and all column headers were converted to lowercase to ensure consistency.

Following the import, the primary focus shifted to managing null values within the dataset. The initial step involved column-level cleaning, where I established two distinct lists of columns slated for removal: `dropNull` and `dropRedundancy`. The `dropNull` list encompassed columns that exhibited excessive nullity, with more than 90% of their values missing across all records. The `dropRedundancy` list comprised columns that provided redundant information, which was otherwise encapsulated by other attributes. Given that the financial forms required a detailed breakdown of reported figures, only the most aggregated data were retained by removing the redundant columns.

To address the remaining null values, record-level cleaning was conducted through simple imputation. Specifically, mean imputation was employed, and the imputed values were rounded to the nearest integer. This rounding was justified as several of the attributes inherently attain only integer values. After these preprocessing steps, the dataset was streamlined from an initial 100 columns to 55, thereby enhancing the manageability and potential predictive accuracy of the ensuing models.

4.1.2 Handling Outliers

The initial approach to managing outliers involved the application of a z-score threshold method. This technique operates by setting a specific z-score limit and excluding all records that exhibit values beyond this predefined range. For this analysis, a threshold of 3 was selected, focusing specifically on the variables 'net income' and 'accounts payable.' This decision was informed by extensive testing across various attributes and different threshold settings, which demonstrated that targeting these particular columns effectively addressed a significant portion of the outliers.

```python
# dealing with a subset of outliers by z-score threshold
def removeOutliers(input_df, column_name, zScoreThreshold):
    mean = input_df[column_name].mean()
    std = input_df[column_name].std()
    z_scores = (input_df[column_name] - mean) / std

    return input_df[(z_scores > -zScoreThreshold) & (z_scores < zScoreThreshold)]

numerical_columns = ['net_income', 'accounts_payable']
for each in numerical_columns:
   df = removeOutliers(fullCostReportdf, each, 3)
```

Listing 1: Function to clean by z-score threshold

Upon completion of the initial outlier management, a subsequent manual review of the dataset was conducted to identify any additional outliers. This examination revealed that the attribute `number of beds` contained suspect outliers. Specifically, eleven nursing facilities reported possessing in excess of 21,000 beds. To address this anomaly, a variable named bedThreshold was defined and set at 21,000. All records exceeding this threshold were subsequently removed from the dataset to ensure the integrity and accuracy of the analysis.

4.1.3 Feature Engineering

This study employs feature engineering techniques to derive additional predictive insights from the dataset's financial metrics. Recognizing that interactions among variables can potentially encapsulate a greater degree of variance than the individual variables in isolation, we have constructed additional features which aim to capture information related to the financial health of the nursing homes. These newly defined features, which capitalize on the interactions between existing variables, are posited to possibly enhance the predictive capacity of the resultant model. Specifically, the following features were engineered and incorporated into the analysis:

1. **Profit Margins:**
   - Gross Profit Margin: (`gross_revenue` - `total_costs`) / `gross_revenue`.
   - Net Profit Margin: `net_income` / `gross_revenue`.
2. **Operational Efficiency:**
   - Total Operating Expense Ratio: `less_total_operating_expense` / `gross_revenue`.
3. **Liquidity Ratios:**
   - Current Ratio: `total_current_assets` / `total_current_liabilities`.
4. **Solvency Ratios:**
   - Debt-to-Equity Ratio: `total_liabilities` / `total_fund_balances`.
5. **Return on Investment (ROI):**
   - ROI: `net_income` / `total_assets`.
6. **Capacity Utilization:**

- Bed Occupancy Rate: `total_days_total` / (`number_of_beds` × 365).

7. **Cost Management:**

   - Cost per Bed: `total_costs` / `number_of_beds`.
   - Salary Costs per Bed: `total_salaries_adjusted` / `number_of_beds`.

In subsequent analyses, not all of the aforementioned features demonstrated statistical significance. This observation is elaborated upon in the later sections of this study, where the impact of each variable on the model's performance is evaluated.

4.1.4 Feature Scaling and Encoding

The preprocessing stage of the analysis involved the application of two normalization techniques to prepare the dataset for subsequent modeling: `StandardScaler()` and `RobustScaler()`. These scaling methods were selected to assess the data's behavior under different normalization conditions and to mitigate the influence of outliers that could skew the results of more sensitive algorithms. The results of these methods will be detailed below in the Analysis section.

Categorical variables were transformed through one-hot encoding to facilitate their integration into the regression models. This process was executed using the `pd.get_dummies()` function from the Pandas library. The specific features subjected to this encoding method were `state_code` and `rural_versus_urban`, enabling the models to handle these categorical inputs effectively.

4.1.5 Exploratory Data Analysis

I generated a set of different bar charts that illustrated various financial features over time. Of these charts, there was a significant change in net income throughout the time period under review. Because of this, and the fact that net income is often a strong indicator of financial health, I selected it as the primary target variable for the analysis.

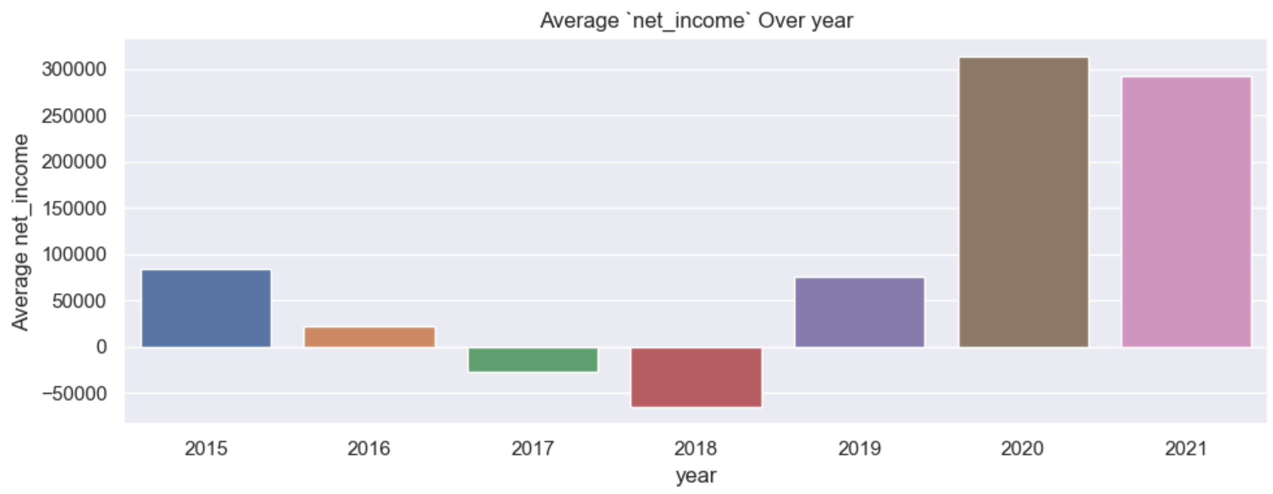


Figure 1: Total `net_income` over time

Net income was also analyzed overtime by state.

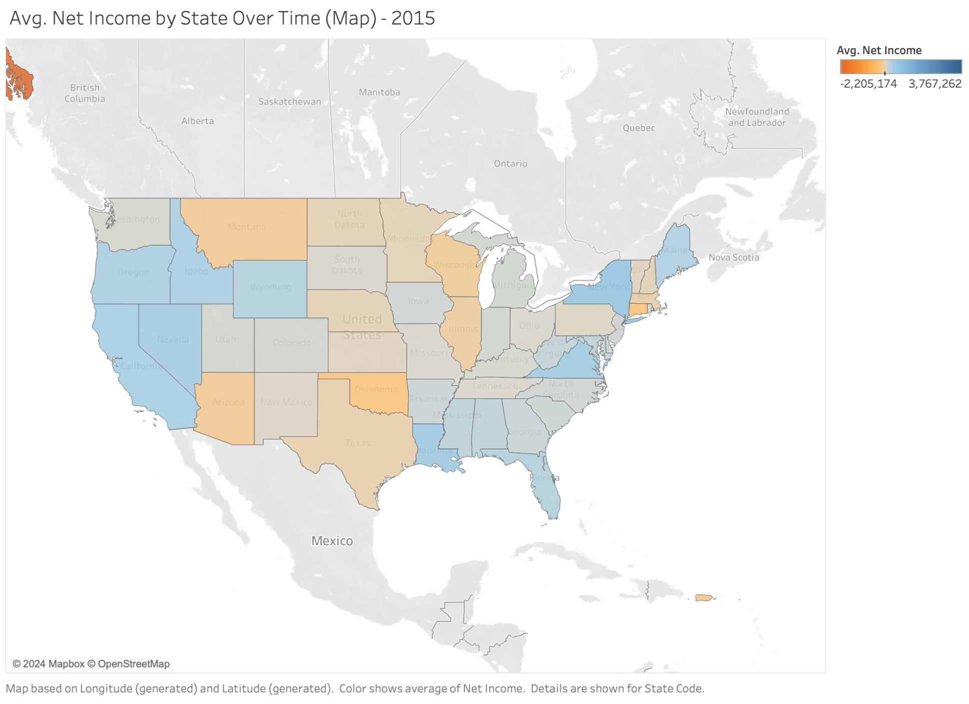


Figure 2: 2015 - Average `net_income` over time by State (Map)

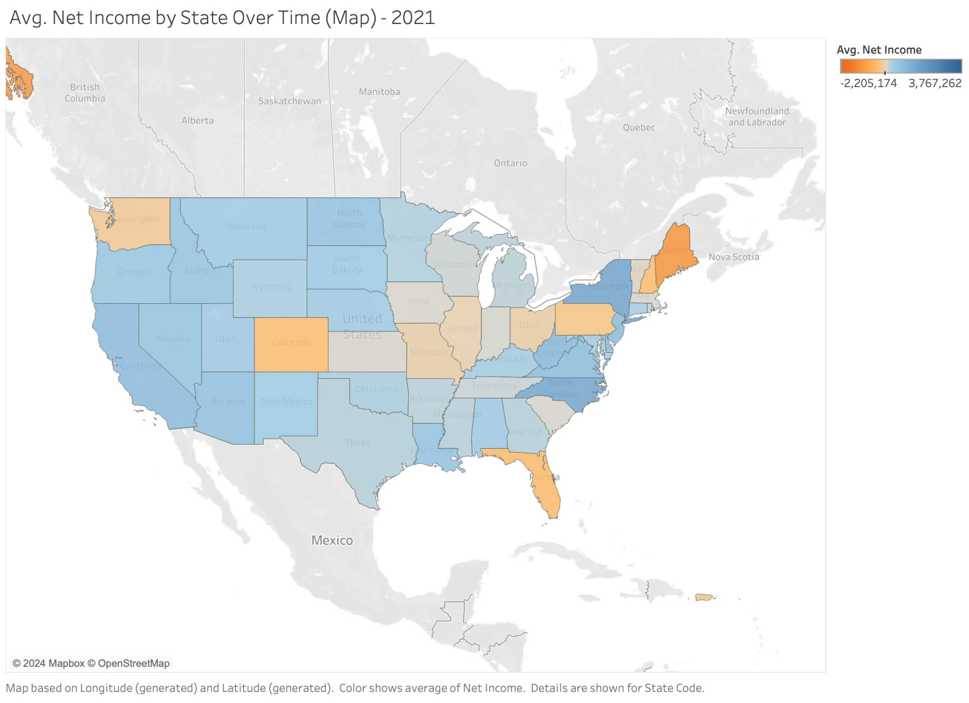


Figure 3: 2021 - Average `net_income` over time by State (Map)

4.2 Provider Information

General information on currently active nursing homes, including number of certified beds, quality measure scores, staffing and other information used in the Five-Star Rating System.

CMS created the Five-Star Quality Rating System to help consumers, their families, and caregivers compare nursing homes more easily and to help identify areas about which you may want to ask questions. The Nursing Home Care Compare web site features a quality rating system that gives each nursing home a rating of between 1 and 5 stars. Nursing homes with 5 stars are considered to have much above average quality and nursing homes with 1 star are considered to have quality much below average. There is one Overall 5-star rating for each nursing home, and separate ratings for health inspections, staffing and quality measures.

4.2.1 Data Loading and Handing Missing Values

Similar to the cost reports, this data set featured data inconsistency issues with the header names between years. The header names were made consistent with the names from 2015, and then all unmatched columns were removed. Any columns with more than 15000 null values were dropped. The data types of the numerical columns was changed to float, and since the provider number acted as the primary key, the data type of the provider number column was made consistent with the cost report data set. Once this was complete, the features net income and net profit margin were joined to the data set and any remaining records that were unable to be properly casted into the correct data type were dropped. Any records that were unable to be matched were also dropped from the dataset.

Because this data consisted of mostly scores on a one through five scale and totals, no outlier detection was applied.

5 Analysis and Findings

Models were were developed and ran against the cost report data, the provider information data, and the COVID data. The sections below detail the model building process and the analysis from each model.

5.1 Model Fit and Comparison for Cost Report Dataset

The analysis incorporated two distinct normalization techniques: `StandardScaler()` and `RobustScaler()`. These scalers were selected to evaluate the robustness and performance of the models under different scaling conditions. The regression algorithms employed in the study included Linear Regression, Lasso Regression, Ridge Regression, Elastic Net Regression, and K-Nearest Neighbors Regression. Each model was systematically applied to the dataset after the application of each scaler to determine the impact on the predictive accuracy. The implementation of these models is detailed below:

```python
def evaluate_regression_models(df, target_column, scaler, test_size=0.1, random_state=123):
    """
    Evaluate various regression models on a given DataFrame with specified scaler.

    Parameters:
    - df : DataFrame containing the data.
    - target_column : string, the name of the column to predict.
    - scaler : Scaler object from sklearn.preprocessing.
    - test_size : float, the proportion of the dataset to include in the test split.
    - random_state : int, random_state is the seed used by the random number generator.
    """

    X = df.drop([target_column], axis=1)
    y = df[target_column]

    # Define numerical features for scaling
    numericalFeatures = X.columns[X.dtypes != 'object']

    # Preprocessor using ColumnTransformer to apply scaling only to numerical features
    preprocessor = ColumnTransformer(
        transformers=[
            ('num', scaler, numericalFeatures)
        ])

    models = {
        'Linear Regression': LinearRegression(),
        'Ridge': Ridge(alpha=20000),
        'Lasso': Lasso(alpha=20000),
        'ElasticNet': ElasticNet(alpha=1.0, l1_ratio=0.5),
        'KNN': KNeighborsRegressor(n_neighbors=2)
    }

    results = pd.DataFrame(index=models.keys(), columns=['Mean Squared Error', 'R-squared', '
    Adjusted R-squared'])

    # Iterate over models, create a full pipeline, and evaluate
    for name, model in models.items():
        pipeline = Pipeline(steps=[('preprocessor', preprocessor),
                                   ('regressor', model)])
        # Split data into train and test sets
        X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=test_size,
    random_state=random_state)
        # Fit the model
        pipeline.fit(X_train, y_train)
        # Predict
        y_pred = pipeline.predict(X_test)
        # Evaluate
        mse = mean_squared_error(y_test, y_pred)
        r2 = r2_score(y_test, y_pred)
        n = len(X_train)
        p = len(X.columns)
        adj_R2 = 1- ((1-r2) * (n-1)/(n-p-1))
        results.loc[name] = [mse, r2, adj_R2]

    print(results)

    # Plotting results
    fig, ax = plt.subplots(1, 2, figsize=(14, 6))
    results.plot(kind='bar', y='Adjusted R-squared', ax=ax[0], color='skyblue', title=f'
    Adjusted R-Squared by Model (Scaler: {scaler})')
    results.plot(kind='bar', y='R-squared', ax=ax[1], color='orange', title=f'R-squared by
    Model (Scaler: {scaler})')
    plt.tight_layout()
    plt.show()
```

Listing 2: Regression model pipeline code

The code above was built in a way such that it would be quick to apply to the different datasets related to this project.

5.1.1 Comparing Model Performance

The performance of various regression models on the dataset is summarized in the figures and tables below. The metrics used for evaluation are the coefficient of determination (R-squared) and adjusted R-squared. The formula used for calculating adjusted r-squared is below. Mean squared error results are contained in the codebase repository, for refenece.

$$R^2_{\text{adjusted}} = 1 - (1 - R^2)\frac{n-1}{n-p-1}$$

where:

- R^2 is the coefficient of determination.
- n is the sample size in the training data.
- p is the number of features in the model.

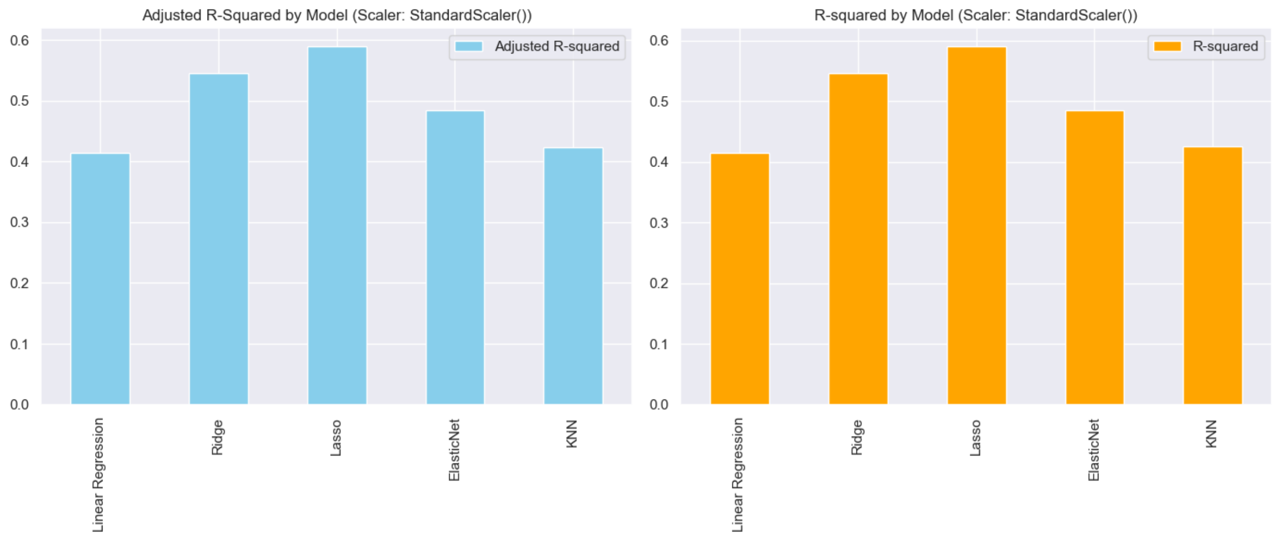


Figure 4: Results for `StandardScaler()`

Model	Adjusted R-squared	R-squared
Linear Regression	0.414	0.416
Ridge	0.545	0.545
Lasso	0.590	0.591
ElasticNet	0.484	0.485
KNN	0.424	0.424

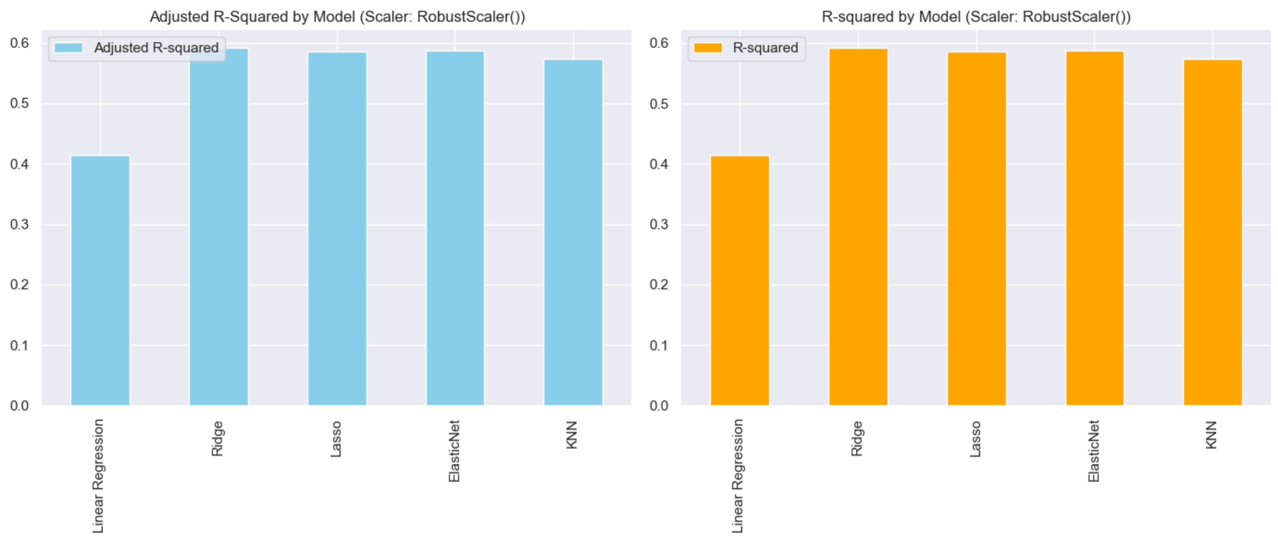


Figure 5: Results for `RobustScaler()`

Model	Adjusted R-squared	R-squared
Linear Regression	0.414	0.415
Ridge	0.592	0.593
Lasso	0.584	0.585
ElasticNet	0.586	0.587
KNN	0.573	0.574

All models perform closely, with Ridge Regression and Lasso performing the best. Additionally, we notice that all models performed better when utilizing `RobustScaler()` as opposed to `StandardScaler()`. Furthermore, given that R-squared and adjusted R-squared are close, we can determine that the model is balanced in terms of complexity and capability.

Because Ridge regression performed the best, this is the model we will go with to predict the net income feature.

5.1.2 Regression Coefficients

In quantitative analysis via Ridge regression, the magnitude and sign of the model's coefficients are indicative of the predictors' association with the target variable—net income of nursing homes. Higher absolute values denote stronger influence, with positive coefficients suggesting an increasing relationship, and negative ones indicating a decreasing effect. Given this, we can determine the most important features by extracting the coefficient values from the model:

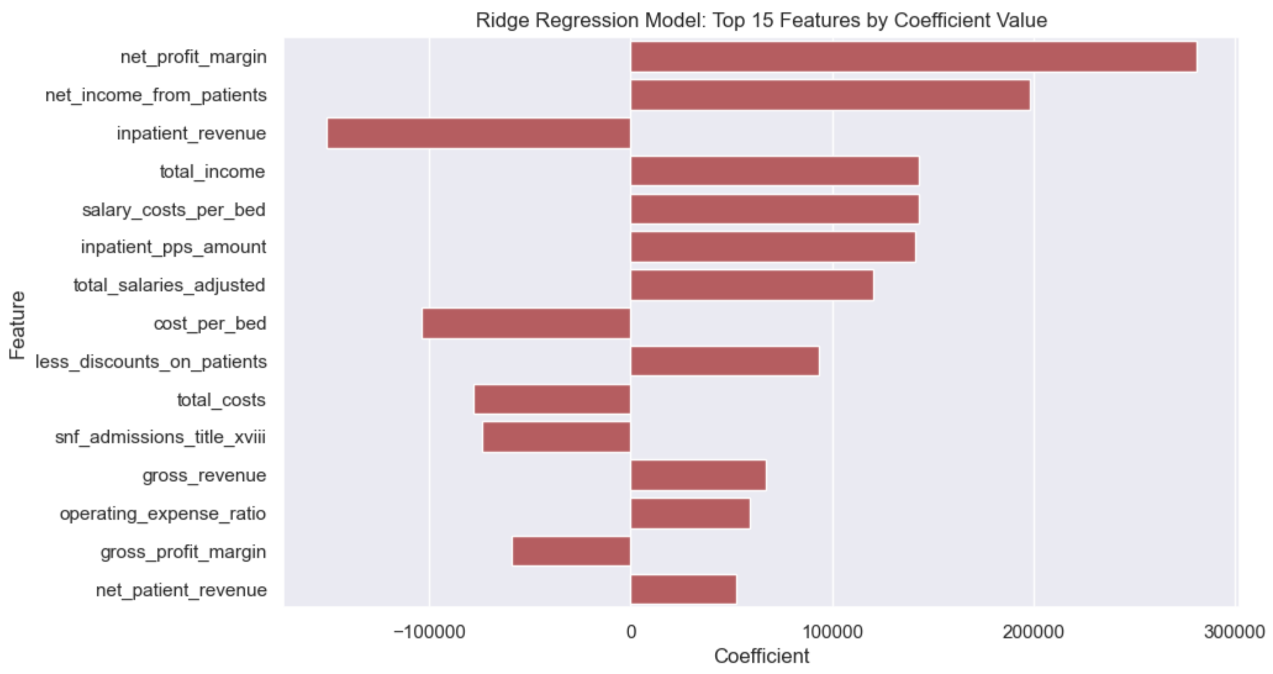


Figure 6: Ridge Regression: Top 15 Coefficient Values by Absolute Value

Feature	Coefficient
net_profit_margin	280896
net_income_from_patients	198654
inpatient_revenue	-151016
total_income	143121
salary_costs_per_bed	142947
inpatient_pps_amount	141518
total_salaries_adjusted	120431
cost_per_bed	-103994
less_discounts_on_patients	93425
total_costs	-78359
snf_admissions_title_xviii	-73877
gross_revenue	67064
operating_expense_ratio	59231
gross_profit_margin	-59231
net_patient_revenue	52355

Several of the features that were engineered show up as important to predict net income. Specifically, net profit margin, salary costs per bed, costs per bed, and operating expense ratio. Among these top features were also the different revenue streams.

5.1.3 Regression Coefficients - States

Although states were less important compared to the other features, we had already found that there are clearly states that perform better than others in terms of net income. By mapping the ridge coefficients of each state, we see a similar picture from the net income maps presented earlier.

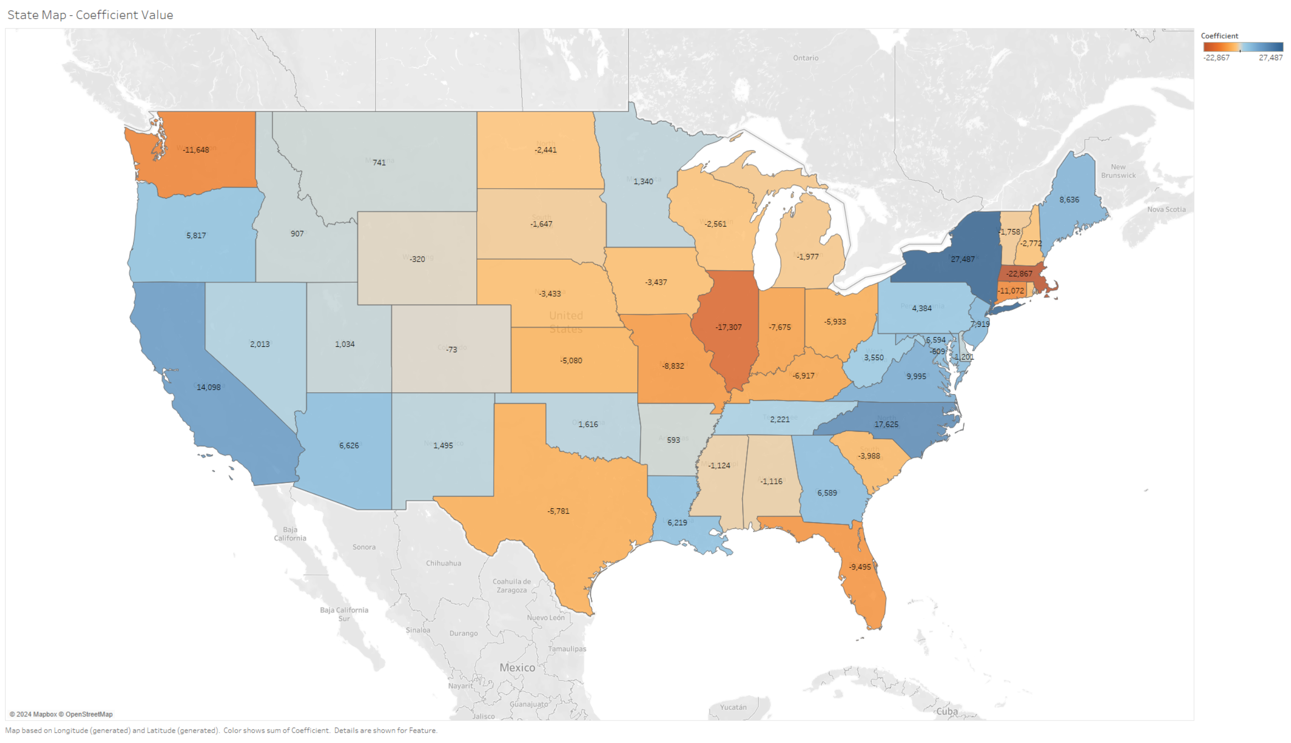


Figure 7: State Map - Ridge Regression Coefficients

Feature	Coefficient
NY	27486.53
NC	17624.84
CA	14097.99
VA	9995.25
ME	8635.73
NJ	7918.63
AZ	6626.17
MD	6593.83
...	...
KY	-6916.69
IN	-7675.09
MO	-8831.85
FL	-9495.18
CT	-11071.54
WA	-11647.73
IL	-17307.36
MA	-22867.15

5.2 Model Fit and Comparison for Provider Data

Using the `evaluate_regression_models()` function, we ran the provider data through the same set of regression models.

6 Discussion

Interpretation of the findings, discussing how they address the project objectives and their implications for the business. This section should also cover any limitations of the analysis and considerations for future research.

7 Recommendations

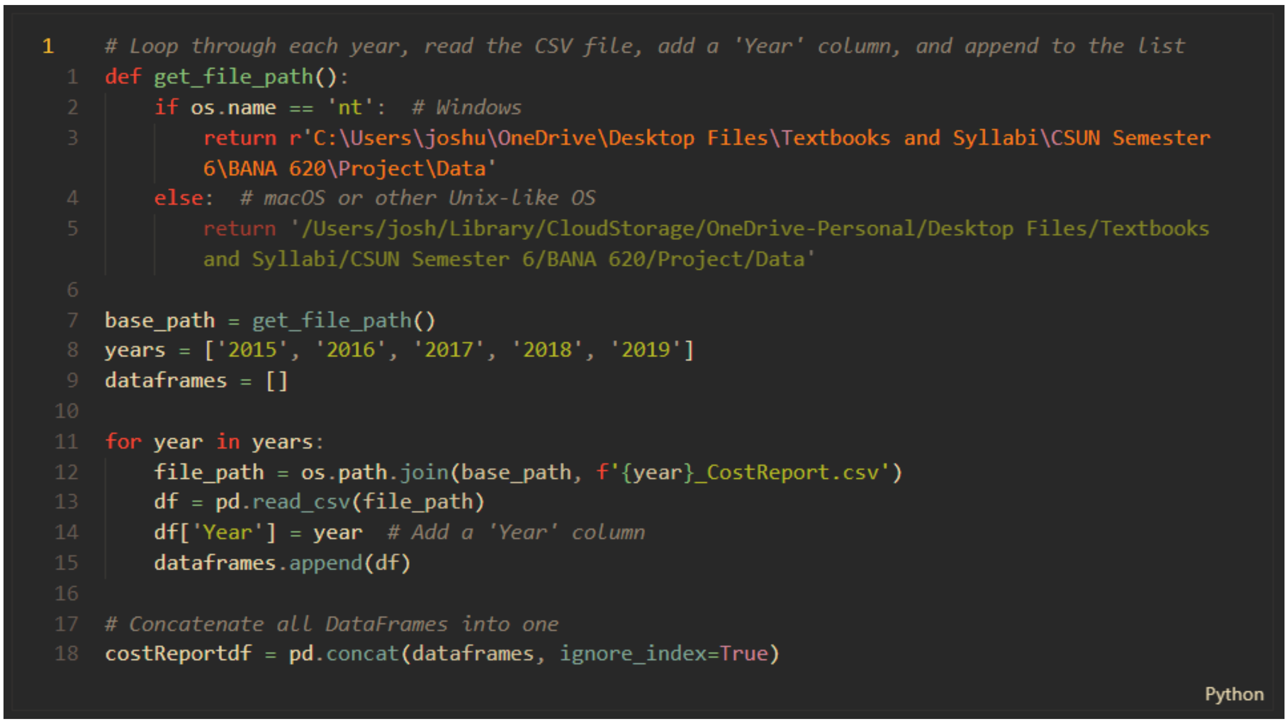
Based on the analysis and findings, provide actionable recommendations for the business. Clearly articulate the expected impact of these recommendations and suggest a plan for implementation.

8 Conclusion

Summarize the key points from the report, reinforcing the value of the findings and recommendations.

9 Appendices

Include any additional material that supports the analysis, such as detailed data tables, code snippets, or extended methodology descriptions.



```python
# Loop through each year, read the CSV file, add a 'Year' column, and append to the list
def get_file_path():
    if os.name == 'nt':  # Windows
        return r'C:\Users\joshu\OneDrive\Desktop Files\Textbooks and Syllabi\CSUN Semester 6\BANA 620\Project\Data'
    else:  # macOS or other Unix-like OS
        return '/Users/josh/Library/CloudStorage/OneDrive-Personal/Desktop Files/Textbooks and Syllabi/CSUN Semester 6/BANA 620/Project/Data'

base_path = get_file_path()
years = ['2015', '2016', '2017', '2018', '2019']
dataframes = []

for year in years:
    file_path = os.path.join(base_path, f'{year}_CostReport.csv')
    df = pd.read_csv(file_path)
    df['Year'] = year  # Add a 'Year' column
    dataframes.append(df)

# Concatenate all DataFrames into one
costReportdf = pd.concat(dataframes, ignore_index=True)
```

Figure 8: Data loading code snippet

10 References

List all sources cited in the report, including data sources, literature, and any external references used in the analysis.