Guided Capstone Project Report

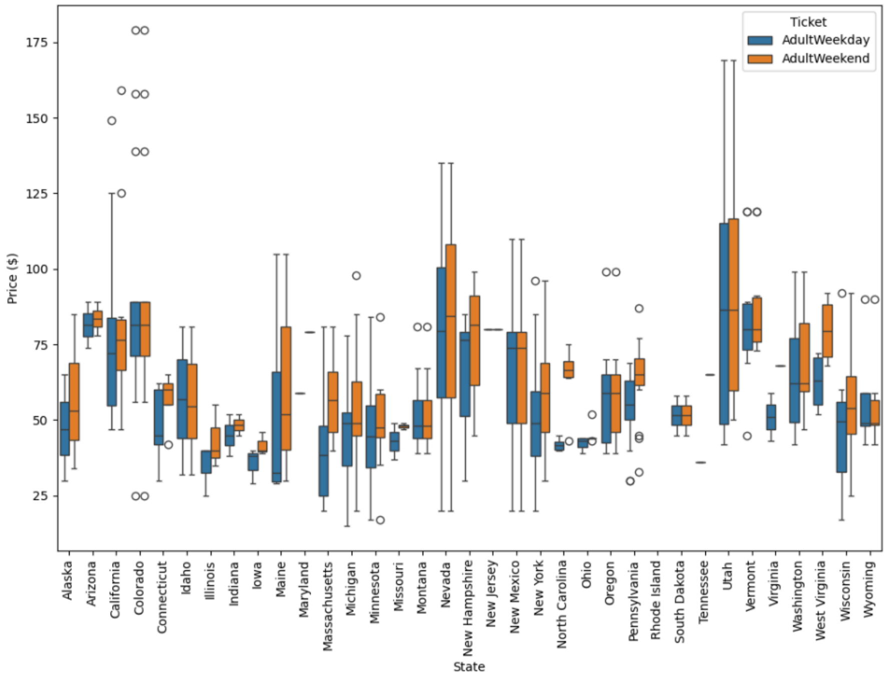
Angeles Olvera 2025

1. Introduction

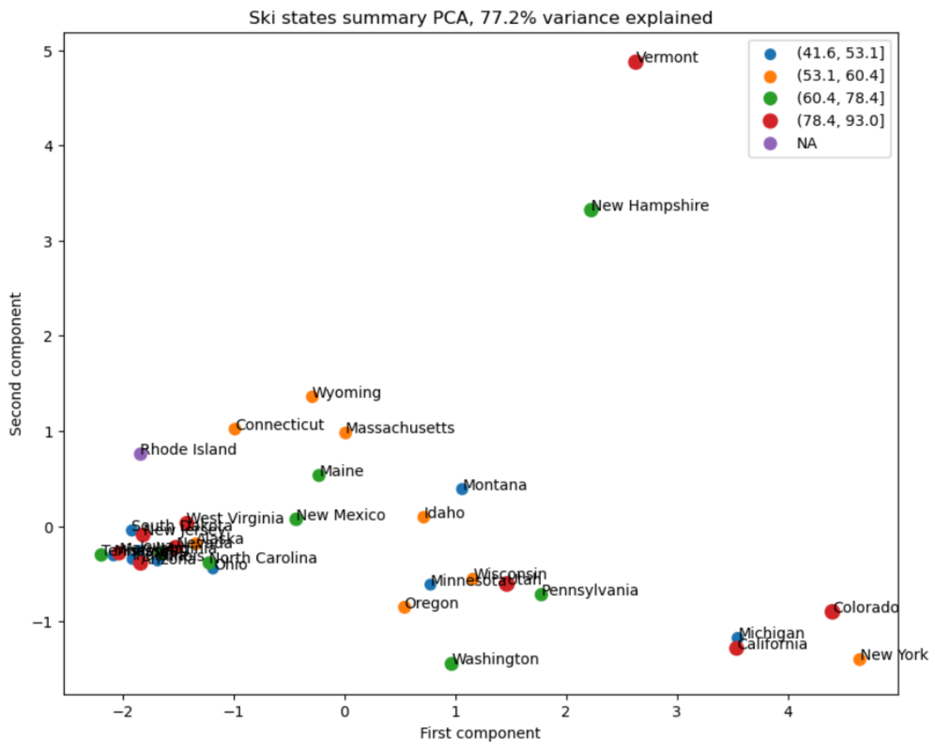
The purpose of this report is to explore how Big Mountain Resort can implement a data-driven business strategy to better utilize its facilities, optimize ticket pricing, reduce costs, and increase profits in the upcoming season. Big Mountain Resort offers access to 105 trails, attracting an average of 350,000 skiers and snowboarders each year. Recently, the resort invested $1,540,000 in a new chair lift to better distribute visitors across the entire park.

2. Data Wrangling and Data Exploration

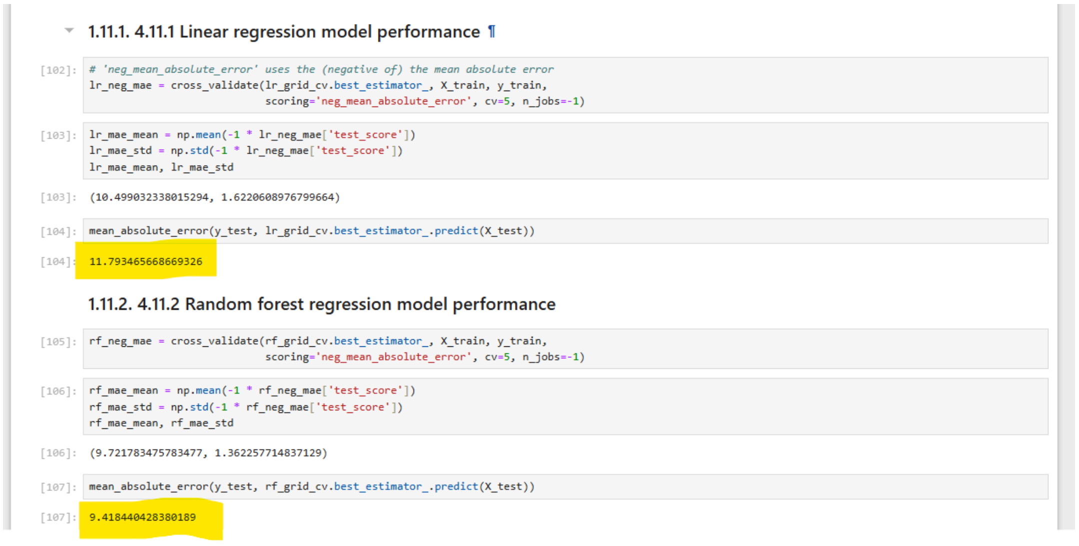
For the data wrangling step, I started by understanding the data, which originally had 330 rows and 27 columns. I specifically pulled out information for Big Data Resort, which had no missing data. I identified 13 columns with missing data and used the .drop() function on the fastEight column, which had 50% missing data, and removed some nonsensical rows. Out of the 27 columns, 3 were categorical: state, region, and resort name. I used a boxplot to investigate resort prices per state and found Montana is not among the highest overall. After further cleaning, I ended up with 277 rows and 35 columns. Additionally, I investigated each state's population to understand if ticket prices are influenced by demand and supply.



For data exploration, I focused on data validation, updating data types and I also worked on feature engineering some new columns that included totals of resorts, skiable areas, resorts per 100 per 100,000 people and 100,000 square miles. When analyzing the number of resorts per 100,000 people and 100,000 square miles I found that most states have 0-25 resorts per 100,000 square miles and 0.0-0.5 resorts per 100,000 people. I then scaled the data to prepare it for Principal Component Analysis (PCA), a technique that simplifies complex datasets by transforming the data into principal components. Then using the explained_variance_ratio_.cumsum allowed me to see that 77% of the data variability is captured by the first two components. The plot shows each point on the plot represents a state, positioned based on its values for PC1 and PC2 The result at the end showed there's no correlation of any way in resort prices per state.



In the Pre-processing and training data step, I began to actually split the data and start getting ready to build a machine learning algorithm. The data was split into a training set and a test set, both of which included their respective target values. Then I did some initial testing prior to building the MLA to see what information was valuable, made sure there were no missing values at all, and removed the categorical features. To make the data transformation simpler and quicker, I used a pipeline that filled the missing values with the SimpleImputer() technique, scaled the data with StandardScaler(), selected the best features with SelectKBest(f_regression), and finally ran the LinearRegression() model. Then I ran a second model option, the Random Forest model. It followed a fairly similar pipeline: it included the SimpleImputer() technique, StandardScaler() technique, and finally the RandomForestRegressor() model. I evaluated both models with MAE, and the model that performed the best was the Random Forest model.



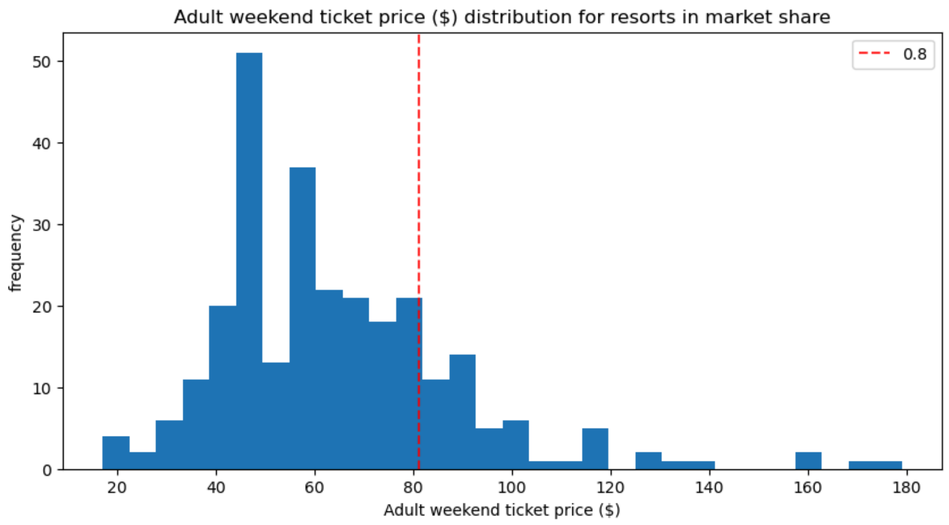
1.11.1. 4.11.1 Linear regression model performance ¶

```
[102]: # 'neg_mean_absolute_error' uses the (negative of) the mean absolute error
       lr_neg_mae = cross_validate(lr_grid_cv.best_estimator_, X_train, y_train,
                                   scoring='neg_mean_absolute_error', cv=5, n_jobs=-1)

[103]: lr_mae_mean = np.mean(-1 * lr_neg_mae['test_score'])
       lr_mae_std = np.std(-1 * lr_neg_mae['test_score'])
       lr_mae_mean, lr_mae_std

[103]: (10.499032338015294, 1.6220608976799664)

[104]: mean_absolute_error(y_test, lr_grid_cv.best_estimator_.predict(X_test))

[104]: 11.793465668669326
```

1.11.2. 4.11.2 Random forest regression model performance

```
[105]: rf_neg_mae = cross_validate(rf_grid_cv.best_estimator_, X_train, y_train,
                                   scoring='neg_mean_absolute_error', cv=5, n_jobs=-1)

[106]: rf_mae_mean = np.mean(-1 * rf_neg_mae['test_score'])
       rf_mae_std = np.std(-1 * rf_neg_mae['test_score'])
       rf_mae_mean, rf_mae_std

[106]: (9.721783475783477, 1.362257714837129)

[107]: mean_absolute_error(y_test, rf_grid_cv.best_estimator_.predict(X_test))

[107]: 9.418440428380189
```

When I passed the data through the model, it predicted that Big Mountain Resort could increase their ticket price from $81 to $97.96. Considering the MAE of $10.36, they have a range of $87.60 to $108.32 to consider. They should definitely consider this, as based on the model's results, Big Mountain Resort is not on the higher end of ticket prices, especially given what they offer.



If they want to be more conservative and conscious of their customers they can increase prices based on specific scenarios. Like if they choose to add a run of 150 feet to the vertical run which would also require adding a new chair lift. This would increase the ticket price by $2.22.

The business leaders of Big Mountain Resort are well-positioned to raise their ticket prices and substantiate this decision based on the available data. While the current analysis provides valuable insights, it would be more precise with additional data, such as maintenance costs and employee wages, to better understand the resort's financial situation.