Big Mountain Resort Report

As part of the analysis I observed the resort price should be modelled at $94.22 as compared with the actual price of $81. The following insights were taken into consideration before coming for a conclusion at each stage of the process.

**◆ Data Wrangling:**

1. First we loaded the data into a readable format and stored it in a variable called “ski\_data” and checked for size of the data which was (330,27).

2. Explored the Resort of Interest which is “Big Mountain Resort” and confirmed the data was present in the raw file.

3. Checked if there were any null values present in any column of data and stored in a variable called “missing”.

4. Calculated the percentage of data missed for each column and realised there was around 16% of data missing for “AdultWeekday” and 15% of data missing for “AdultWeekend”.

5. Checked for Duplicates with resort names as base and found one resort i.e., Crystal Mountain. Upon further investigation with other columns names confirmed there was no duplicate entry recorded.

6. Calculated averages for each state for the required columns “AdultWeekday” and “AdultWeekend” using mean function and stored in a variable called “state\_price\_means”.

7. Created a “barh” plot to show average ticket prices varies from state to state in Decreasing order titled “Average ticket price by state” to get more insights into the differences in the distributions between states.

8. Transformed the data into a single column for price with a new categorical column that represent the ticket type into a variable “ticket\_prices”.

9. Created a “box” plot to check where majority of points are falling and noticed most prices appear to lie in broad band from around $25 - $100. Few states like California, Colorado and Utah were out of range.

10. Calculated the percentage of missing prices for “AdultWeekend” and “AdultWeekday” and found just over 82% of resorts have no missing ticket price, 3% are missing one value, and 14% are missing both.

11. As per the recommendations provided the 14% of missing values in both the columns were dropped.

12. There were few wrong values which were rectified like “Silverton Mountain resort” SkiableTerrain\_ac value noted as 26819 which in actual was 1819 and “Heavenly Mountain resort” SkiableTerrain\_ac value

noted as 4800 instead of 2880.

13. There’s one column “fastEight” in which half the values were missing and all the other values are zero. As there is no essential information, dropped the entire column.

14. After analysing the “yearsOpen” column, found two outliers which were 104 years and 2019 years. As we can certainly state that no resort will have been open for 2019 years and not much information was provided,

dropped the entire row and considered the other as reasonable outlier.

15. Created new table with groupby function and extracted the information about ‘resorts\_per\_state’, ‘state\_total\_skiable-area\_ac’, ‘state\_total\_days\_open’, ‘state\_total\_terrain\_parks’, ‘state\_total\_nightskiing\_ac’ based

on state and stored in a variable called “state\_summary” to clearly perform the aggregations.

16. After deleting few rows and a column created a histogram for each column subplots\_adjust and found few skewed distributions like “fastQuads”, “fastSixes”, “trams” which needs to be verified further during analysis.

17. Imported one more dataset with variable “usa\_states” which contains all the states names, population, state\_area, “capita” etc.

18. Extracted few columns like “state”, “state\_population”, “state-area\_sq\_miles” and stored in a variable “usa\_states\_sub”.

19. Checked to see if there are any states missing in the “state\_sumamry” with “ usa\_states\_sub” and found few values but upon further analysis all states were included in the set.

20. Merged both tables “state\_summary” and “usa\_states\_sub” based on state and created a Scatter plot to find the relationship between Weekday and Weekend prices and found majority of prices are equal and are

restricted under $100.

21. Using .loc accessor filtered “Montana” state with Weekday and Weekend mean prices and found 4 AdultWeekend and 7 AdultWeekday prices. As there is no difference between these two prices and more

AdultWeekday prices were missing, decided to drop the column.

22. After all the changes in the data, the final shape of the new data was (277,25) and stored to local repository under name “ski\_data\_cleaned.csv”.

23. Newly created table “state\_summary” was also stored in local repository under “state\_summary.csv” which we will use in the next stage “**Exploratory Data Analysis**”.

**◆ Exploratory Data Analysis:**

1. First we loaded the both data sets “ski\_data\_cleaned.csv” and “state\_summary.csv”.

2. Reset the index of state\_summary from numbers to state and stored in a variable called “state\_sumamry\_newind”.

3. Sorted values in Descending order to find the top 5 values of states with “Highest Area”, “Highest Population”, “Highest Resorts per state”, “Total skiable area”, “Total Night Skiing area”, “Total days open” to find some insights.

4. Calculated the measures of resort density per 100K capita and sqkms and stored in new variables “state\_summary['resorts\_per\_100kcapita’]” and “state\_summary['resorts\_per\_100ksq\_mile']” and dropped the absolute population and state size columns. Created Histograms on these two features and found a Skewed Distribution towards right.

5. Resorted the values in Descending order for the above two features and filtered the top 5 states and found Montana in fourth position in terms of “population” and not in resorts per 100kq\_mile.

6. After constructing some features derived from statistics for each of the states, we used “**Principle Component Analysis(PCA)**” to find linear combinations of the original features and to order them by variance and to visualise data in a lower dimension.

6.1: Scaled the state\_summary data by setting the state as index and stored index values in a different variable “state\_summary\_index” and stored all the columns names under different variable “state\_summary\_columns” and scaled the data.

6.2: After Scaling data checked the mean which adds up to zero and standard deviation for each feature more than one. Upon providing the biased estimator (ddof=0) standard deviation became 1.

6.3: Performed PCA transformation on the scaled data and created a line plot to visualize cumulative explained variance ratio with number of components.

6.4: This plot explians that first two components account for 75% of variance and first four account for over 95%.

6.5: Created Scatter plots of first two components to see the variance between the states by plotting First Component on x-axis and Second Component on y-axis.

6.6: Used groupby function on ski\_data to get the AdultWeekend prices and calculated the mean for each state.

6.7: used PCA values of state\_summary data and merged into a new DataFrame with state name as index, and AdultWeekend mean values and defined it with new variable “pca\_df”.

6.8: We then created one more column on pca\_df and named it “Quartile” and calculated the quartile based on “Quantile based Discritization Fuction”and to make sure data is cleaned enough checked for any missing values. Found one row with NaN values and filled the weekend price with mean values and for quartile values converted into categorical values ‘NA’ instead of numpy NaN values.

6.9: Created Scatter plot again with loop which iterates over each quartile and plots the points in the same quartile group with points coloured by quartile and sized by ticket price.

6.10:The Scatter Plots created using matplotlib library do not make any pattern with price, so opted for seaborn scatterplot to visually explore multiple features and to make use of colours in ascending order and for more transparent view.

7. After visualising the data we found that two states “New Hampshire ” and “Vermont” have particularly large values and these put them more than 3 Standard Deviations from the mean.

8. To explore the resort level data more, merged state\_summary data with ski\_data to engineer few features.

9. Calculated ratios for “Skiable\_Area”,”Number of days Open”, “Terrain\_parks\_count”, ”Night\_skiing\_area” with context of its state to understand what share of states skiing assets is accounted by each resort. Once after calculating the actual ratios, dropped those state columns and replaced with new columns of ratio values. Then created a seaborn heatmap to gain high level view relationship among the features.

10.Created scatterplots to check how ticket price varies with other different features and observed few things.

10.1: There appears a positive correlation with “vertical\_drop”, “fast\_quads”, “Runs”, “total\_chairs”.

10.2: Resorts\_per\_100kcapita tells us when the value is low, there is a variability in ticket\_price, although it is capable of going high. Ticket price may drop when the resorts increases but could climb with the number of resorts serving population indicates a popular skiing area with more demand.

11. Created ratios of chairs to runs as it seems logical that this ratio would inform you how quickly people can get to their next ski and created scatterplots. We found that the fewer chairs we serve, the ticket price would be higher but may serves less visitors. Data on number of visitors per year also helps data find a relationship.

12.Created ratios of fast\_quads with runs and Skiable\_Terrain and plotted them to find the relationship and it appears that having no fast quads may limit the ticket price, but if your resort covers a wide area then getting a small number of fast quads may be beneficial to ticket price.

13. After making all the changes to the columns and data, the new data is stored in a different file in local repository under name “**ski\_data\_step3\_features(EDA).csv**” which we will use in next stage “**Preprocessing and Training**”.

◆ **Preprocessing and Training:**

1. First we Loaded the dataset “ski\_data\_step3\_features(EDA).csv”

2. Divided data on 70/30 ratio to train/test data to generate a model.

3. Calculated mean value on train data which is 63.811088082901556.

4. Calculated mean value using sklearn DummyRegressor to check if it returns same value 63.811088082901556 and got same results.

5. Calculated R\_Squared values to verify how close predicted values are close to the actual values. The values appeared are ‘0.0’ for train data and ‘-0.0031235200417911724’ for test data.

6. Calculated Mean\_Absolute\_Error to get an idea on how close our predictions are to the true values. The values appeared are ’17.923463717146785’ for train data and ’19.136142081278486’ for test data which essentially tells us

that we might expect to be off by around $19 if we guessed ticket price based on an average of known values.

7. Calculated Mean\_Squared\_Error to predict the values. The values appeared were ‘614.1334096969057’ for train data and ‘581.4365441953481’ for test data which highlights that results for MSE on test data is better than train data.

8. As there are many missing values in the data, we imputed missing values using median for further analysis. Then Scaled the data using ‘StandardScaler’ and tried to train the model once again to predict the model performance.

8.1: R\_Square value came as (0.8177988515690604, 0.7209725843435144) for train and test.

8.2: MAE value came as (9.407020118581315, 9.407020118581315) for train and test.

8.3: MSE value came as (161.73156451192276, 161.73156451192276) for train and test.

8.4: This model expect to estimate ticket price within $9 as compared by using just the average.

9. We imputed missing values using mean to find the model performance as compared with median.

9.1: R\_Square value came as (0.8170154093990025, 0.7163814716959965) for train and test.

9.2: MAE value came as (8.536884040670973, 9.416375625789271) for train and test.

9.3: MSE value came as (112.37695054778276, 164.39269309524335) for train and test.

9.4: These results did not show much difference when imputed with median or mean.

10. Created Pipeline to store the entire process from imputation to training the model and cross checked if the values gained earlier were same as compared with a different process. The values arrived were same.

11. Redefined the Pipeline with new feature SelectKBest feature which selects the best features and trained the model again.

11.1: R\_Square value came as (0.7674914326052744, 0.6259877354190837) for train and test.

11.2: MAE value came as (9.501495079727484, 11.201830190332052) for train and test.

11.3: These values were far less accurate as compared with the earlier values. Changed the default value of SelectKBest from 10 to 15 and trained the model.

11.3.1: R\_Square value came as (0.7924096060483825, 0.6376199973170795) for train and test.

11.3.2: MAE value came as (9.211767769307114, 10.488246867294357) for train and test.

12. Assessed performance using Cross\_validate by specifying the number of blocks as cv=5 and analysed the scores. The test scores are [0.63760862, 0.72831381, 0.74443537, 0.5487915 , 0.50441472]. Calculated the mean and Std for these test scores and got values (0.6327128053007863, 0.09502487849877697) which highlights the assessing model performance.

13. Performed Hyperparameter Search using GridSearchCV by passing K best feature as grid\_params to find out the best parameter which helps to increase the accuracy.

14. Created RandomForestRegressor model and inserted into pipeline and calculated the test scores which are [0.6877383 , 0.77987845, 0.77753583, 0.62190924, 0.61794573] with mean and std (0.6970015087370294, 0.07117420491358241).

15. After performing the GridSearchCV by specifying grid\_params with certain conditions and test the model once agin. The test score values are [0.68961442, 0.79593861, 0.77128447, 0.62254707, 0.66142192] with mean and std

(0.7081612973109424, 0.06564715369819316). The CV results are marginal. Random Forest has many more hyperparameters we could tune to increase the margin.

16. Created a Box Plot to check the best regressor feature importances and found top four features(fastQuads, Runs, Snow making\_ac, vertical\_drop) are in common with the linear model.

17. As part of final model selection

17.1: calculated Linear Regression model performance and the values obtained for mean and std are (10.499032338015294, 1.6220608976799664) with MAE of ’11.793465668669327’.

17.2: calculated RandomForest model performance and the values obtained for mean and std are (9.659539811066127, 1.3496029127071227) with MAE of ‘9.495505500261919’.

17.3: RandomForest has a lower cross-validation MAE by almost $1 which exhibits less variability.Verifying performance on the test set produces performance consistent with the cross-validation results.

18. Performed data quantity assessment using learning\_curve function by randomly specifying few values to the train\_sizes and calculated the train\_scores and test\_scores mean and std and plot it on a errorbar. This plot depicts

that as the size increases, theres an initial improvement in model scores but leveled-off around sample size of 40-50.

19. As the final step, the best model which gives more accuracy with best mean and std are stored as ‘ski\_resort\_pricing\_model.pkl’ which we will use in the final stage ‘**Modeling**’.

◆ **Modelling:**

1. Loaded the model “ski\_resort\_pricing\_model.pkl” and loaded the data “ski\_data\_step3\_features(EDA).csv”

2. Fit model on data and cross validate it. The test scores are [-12.10337215, -9.28661397, -11.41279578, -8.06408169, -11.05864559] and the mae\_mean and mae\_std values at (10.3851018351214, 1.487015739861695).

3. Calculated Expected Big Mountain Ticket price from the model which values at $94.22 where as the actual price is $81 even with an mean of $10.39 which suggests there is a scope for increase.

4. Created a function which takes the important KBest feature as input and plots the histogram of the value of that feature . It marks a axvline showing the target feature Big Mountain Resort where exactly the value sits in the distribution.

5. Business has shorlisted 4 scenarios which may help to sense how facilities support a given ticket price and also some additional information of expected number of visitors over the season is 350,000 and on average visitors ski for 5 days and also assuming that the provided data include additional lift that Big Mountain has installed.

5.1: Scenario 1 suggests to close up to 10 of the least used runs. This explains that closing one run makes no difference in price but closing 2 and 3 reduces support for ticket price and if they close run 3 they may close 4 and 5 as well as there is no further loss in ticket price. Increasing the closure down to 6 or more leads to a large drop.

5.2: Scenario 2 suggests to increase vertical drop by 150 ft and installing additional air lift. This Scenario helps increase support for ticket price by $1.99 and over the season this could be expected to amount for $3474638.

5.3: Scenario 3 suggests to repeat scenario 2 but adding 2 acres of snow making. This Scenario helps increase support for ticket price by $1.99 and over the season this could be expected to amount for $3474638 which makes no difference in the predicted price.

5.4: Scenario 4 suggests for increasing the longest runs by.2 miles and adding 4 acres of snow making capability. This scenario creates no difference though longest run feature was used in linear model, random forest model only has longest run way down in the feature importance list.

6. The model calculated the price assuming other resorts accurately set their prices according to market supports. Its reasonable to expect some resorts might have overpriced or underpriced the price value data. We know nothing about operating cost. We could use this kind of information to assess the more accurate price for Big Mountain Resort.