Byrnecut Australia Advance Predictor

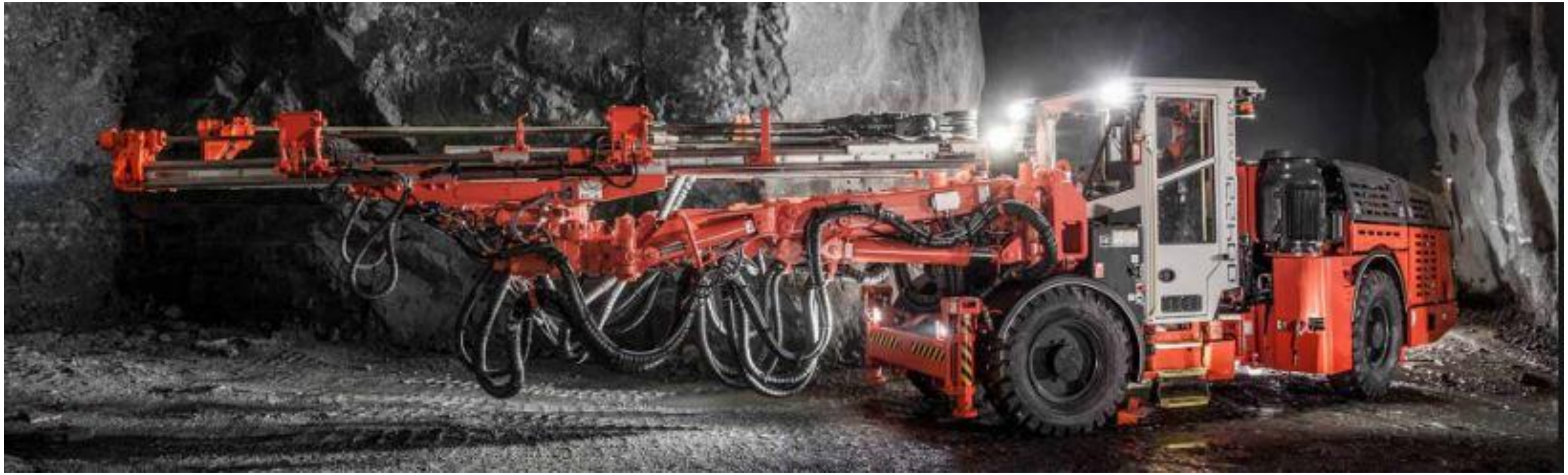
November 2022

Blake Elieff

BYRNECUT
Australia

Summary

Analysis of the JumboData dataset and building of a model that can predict Jumbo advance based off of different features



Outline

- Definitions
- Business Problem
- Data & Methods
- Results
- Conclusion

Definitions

- **Jumbo** - A piece of machinery used to drill holes to put explosives in to break down dirt underground, shown in the photo on the summary page. This machine also installs ground support to protect mining personnel underground
- **Advance** - How far we progress underground, the **Jumbo** drills holes in the face of the rock which is then filled with explosives. Once the explosives are set off we call this a 'cut'. Each 'cut' we take, is approximately 4m of advance
- **Front Liner** - A front liner is our fully trained/highly skilled operators on the Jumbo
- **Trainee** - A new jumbo operator, not as experienced so not as productive as a Front Liner
- **DRM** - Drill Metres, the sum of ream and face holes drilled

BYRNECUT Australia

Business Problem

A model is required that is able to predict the advance for a week based on provided values for multiple features. This model is needed to answer the following questions:

- If we gain an extra heading, what effect will it have on advance for the week
- If we replace one of our frontliners with a trainee what effect will it have on advance for the week

BYRNECUT
Australia

Data and Methods

- 1 Dataset (JumboData.csv)
- Used Standard scaling, One hot encoding, train test split, OLS

BYRNECUT
Australia

Model

- This is the function that I was able to produce to work with my model
- To test out my model, I use feature values from our current weekly schedule

```python
def predict_advance(small_headings,big_headings,rigs,frontliners,trainees,start_date):
    '''

    small_headings = the quantity of small headings in the weekly schedule
    big_headings = the quantity of small headings in the weekly schedule
    rigs = the quantity of jumbos we have this week
    frontliners = the quantity of frontline operators we have this week
    trainees = the quantity of trainee operators we have this week
    start_date = the start date of our week in format 'DD/MM/YY'

    '''

    #Import datetime module
    from datetime import datetime

    #Determine a non scaled amount of weekly headings
    weekly_headings_ns = small_headings + big_headings

    #Determine non scaled values for amount of rigs
    if rigs == 4:
        four_rigs_ns = 1
    else: four_rigs_ns = 0
    if rigs == 3:
        three_rigs_ns = 1
    else: three_rigs_ns = 0

    #Determine non scaled percentage of trainees and small headinsg
    perc_trainee_ns = trainees / (frontliners + trainees)
    perc_small_ns = small_headings / (small_headings + big_headings)

    #Determine a non scaled amount of days from the start of contract to our current date
    date_delta1_ns = (datetime.strptime(start_date,'%d/%m/%y') - datetime.strptime('5/7/17','%d/%m/%y')).days
    #Do the same for the  next 6 days
    date_delta2_ns = date_delta1_ns + 1
    date_delta3_ns = date_delta2_ns + 1
    date_delta4_ns = date_delta3_ns + 1
    date_delta5_ns = date_delta4_ns + 1
    date_delta6_ns = date_delta5_ns + 1
    date_delta7_ns = date_delta6_ns + 1

    #Scale all of the date deltas
    date_delta1 = std_scale(date_delta1_ns, mean_date, std_date)
    date_delta2 = std_scale(date_delta2_ns, mean_date, std_date)
    date_delta3 = std_scale(date_delta3_ns, mean_date, std_date)
    date_delta4 = std_scale(date_delta4_ns, mean_date, std_date)
    date_delta5 = std_scale(date_delta5_ns, mean_date, std_date)
    date_delta6 = std_scale(date_delta6_ns, mean_date, std_date)
    date_delta7 = std_scale(date_delta7_ns, mean_date, std_date)

    #scale every other feature
    weekly_headings = std_scale(weekly_headings_ns, mean_hed, std_hed)
    perc_trainee = std_scale(perc_trainee_ns, mean_train, std_train)
    perc_small = std_scale(perc_small_ns, mean_sml, std_sml)
    three_rigs = std_scale(three_rigs_ns, mean_3, std_3)
    four_rigs = std_scale(four_rigs_ns, mean_4, std_4)

    #Run the prediction function for the model for each date delta
    X1 = [[date_delta1,weekly_headings,perc_trainee,perc_small,three_rigs,four_rigs]]
    adv1 = final_model2.predict(X1)
    X2 = [[date_delta2,weekly_headings,perc_trainee,perc_small,three_rigs,four_rigs]]
    adv2 = final_model2.predict(X2)
    X3 = [[date_delta3,weekly_headings,perc_trainee,perc_small,three_rigs,four_rigs]]
    adv3 = final_model2.predict(X3)
    X4 = [[date_delta4,weekly_headings,perc_trainee,perc_small,three_rigs,four_rigs]]
    adv4 = final_model2.predict(X4)
    X5 = [[date_delta5,weekly_headings,perc_trainee,perc_small,three_rigs,four_rigs]]
    adv5 = final_model2.predict(X5)
    X6 = [[date_delta6,weekly_headings,perc_trainee,perc_small,three_rigs,four_rigs]]
    adv6 = final_model2.predict(X6)
    X7 = [[date_delta7,weekly_headings,perc_trainee,perc_small,three_rigs,four_rigs]]
    adv7 = final_model2.predict(X7)

    #Sum up the advances for the week
    advance = adv1 + adv2 + adv3 + adv4 + adv5 + adv6 + adv7

    #print
    print(round(float(advance),1),'metres')
```

BYRNECUT
Australia

Results

```
# Lets use the feature values from the current week's schedule..
predict_advance(small_headings=8,
               big_headings=8,
                rigs=4,
                frontliners=8,
                trainees=0,
               start_date='2/11/22')
```

```
233.5 metres
```

So our model has predicted 233.5m for the week. thats an average of 33.36 a day

This week so far (in real life) we are averaging 33.73m a day

This is a great result!

BYRNECUT
Australia

Results

- If there is an extra small heading, expect to gain 6.3m

```
#using the current week, what if we had an extra small heading?
predict_advance(small_headings=9,
               big_headings=8,
                rigs=4,
                frontliners=8,
                trainees=0,
               start_date='2/11/22')
```

```
240.8 metres
```

Results

- If there is an extra large heading, expect to gain 5.4m

```
#using the current week, what if we had an extra large heading?
predict_advance(small_headings=8,
               big_headings=9,
                rigs=4,
                frontliners=8,
                trainees=0,
               start_date='2/11/22')
```

```
238.9 metres
```

Results

- If a front line operator is swapped out for a trainee, expect to lose 12m

```
#using the current week, what if we replaced a front liner with a trainee?
predict_advance(small_headings=8,
               big_headings=8,
                rigs=4,
                frontliners=7,
                trainees=1,
               start_date='2/11/22')
```

```
221.5 metres
```

Conclusion

- From this analysis and model production. I can conclude that Byrnecut should use the quantity of rigs, level of operators, date, size of the headings and quantity of the headings available to determine the expected advance for each week.

- This model is able to be used to determine the expected advance for each week quite accurately. Through conducting real world testing this week the prediction is within 0.5m per day.

BYRNECUT
Australia

Thank You

Email: blakeelieff@hotmail.com
Github: @elieff