

Mar 1, 2024

- Class updates next week:
 - Show the time-series model performance
 - Live demo of the API for model 2?

Welcome aboard, friend! 🙌

We want to get to know you better. Let's start with some 'What would you do' questions.

If you received an unexpected \$100 in cash, what would you do?

- ☒ Save it in cash at home
- ☐ Invest it
- ☐ Spend it right away
- ☐ Donate it

How many months' worth of expenses do you feel comfortable having in your emergency fund?

- ☒ Less than 1 month
- ☐ 1-3 months
- ☐ 3-6 months
- ☐ More than 6 months

When thinking about investing, which statement best describes your approach?

- ☒ I prefer safe investments with low returns
- ☐ I am willing to take moderate risks for moderate returns
- ☐ I am focused on high-risk, high-reward investments
- ☐ I am not interested in investing

Which of the following is your top financial priority right now?

- ☒ Paying off debt
- ☐ Saving for retirement
- ☐ Building an emergency fund
- ☐ Saving for a major purchase (e.g., home, car)

What is your most important long-term financial goal?

- ☒ Achieving financial freedom
- ☐ Buying a home
- ☐ Saving for children's education

How can I help you?

why is saving so hard for me?

Ask Agent!

Bedrock answer

It seems saving is hard for you for a few key reasons:

1. Your top priority of paying off debt requires allocating income to debt repayment each month rather than savings.
2. Your emergency fund is currently very low, so building it up requires shifting more funds away from debt repayment and towards short-term savings.
3. Reaching your long-term goal of financial freedom will require both paying down liabilities and increasing long-term savings simultaneously, which can be challenging without automation.

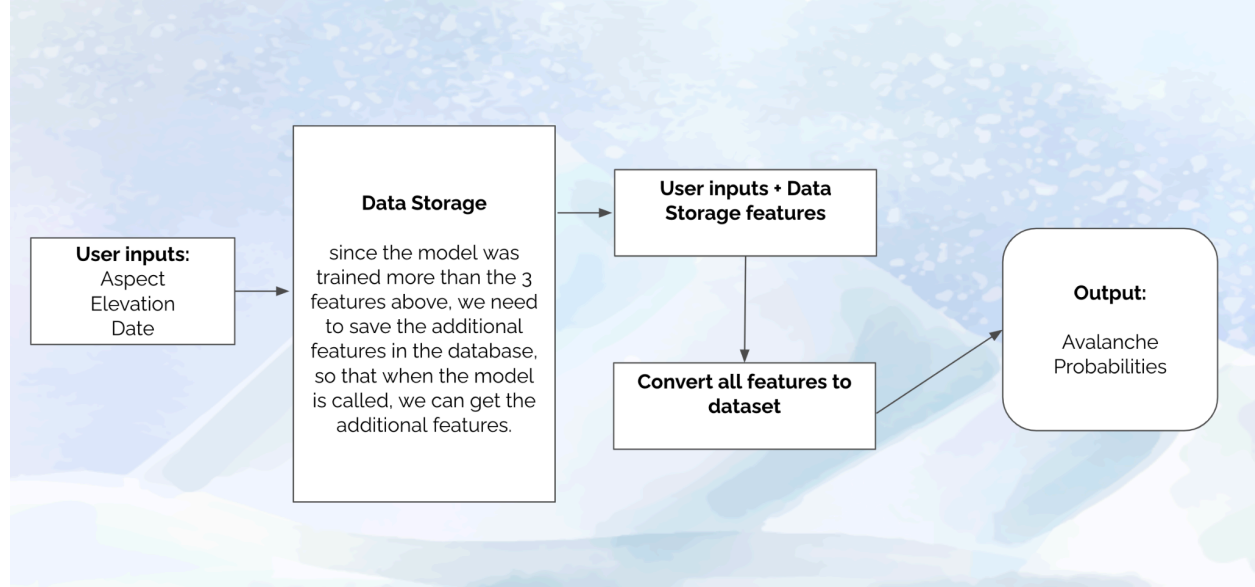
To make saving easier, I recommend automating savings transfers as discussed previously to build saving into your monthly routine without much effort. Paying down the highest interest debt can also free up funds to allocate more to savings each month going forward.

Time taken Bedrock to process: 11.55 seconds

OpenAI answer

Based on your behavior and financial status, you seem to prioritize debt repayment over saving, possibly impacting your ability to save. It's essential to create a budget that includes both debt repayment and savings to achieve your financial goals

Model 2 API Workflow



- **Part 2 requirements in detail (Ravi):** This tab requires users to input information(refer to input requirements), find additional features (refer to requirements), take the inputs and go through a model (refer to base model file), produce an output (output requirements).
 - **Input Requirements:**
 - User inputs, the UI requires users to input: aspects, elevation, and date

- Aspects: create a drop-down for the available aspects:
 - N
 - NE
 - E
 - SE
 - S
 - SW
 - W
 - NW
- Elevation: create a drop-down for the available elevation:
 - Below Treeline
 - Near Treeline
 - Above Treeline
- Date (based on our discussion):
 - we have to forecast data (features) until 1/30/2024, we saved our features in the table so that I can call the features from the same table. ([combined_date.csv](#)).
- **Additional features:** since the model was trained more than the 3 features above, we need to save the additional features in the database, so that when the model is called, we can get the additional features:
 - The additional features table: final_data.csv
 - To load these features:

```
[11] final_data = pd.read_csv(
    f"(DATA_PATH_FINAL)/final_data.csv", keep_default_na=False).dropna(subset=['combined_terrain_aspects', 'combined_terrain_elevations'],
    how="any"
)
```

- **User inputs + Data Storage features: Demo**
<https://colab.research.google.com/drive/1U-aiT2ALaHaCMBh9DM0DTTDpVfuCrh6R?authuser=2#scrollTo=D2PPjqOUCw7nb>
 - Save the additional features (shape should be xx, 3157)
 - Take the inputs from users (shape should be xx, 3)
 - Concat them together (on = 'date_time' to find the 3157 features in addition to the user inputs: aspect and elevation)
 - **Convert the concat df to dataset function:**

```
def df_to_dataset(dataframe, shuffle=True, batch_size=32, target_col="target"):
    df = dataframe.copy()
    labels = df[target_col]
    df = {key: value.tolist() for key, value in dataframe.items()}
    ds = tf.data.Dataset.from_tensor_slices((dict(df), labels))
    if shuffle:
        ds = ds.shuffle(buffer_size=len(dataframe))
    ds = ds.batch(batch_size)
    ds = ds.prefetch(batch_size)
    return ds
```

- **Updated model file:**

- The model file is located in [gdrive file](#): tf_rand_val_nn_v1
- To load model:

```
[4] drive.mount("/content/gdrive")
Mounted at /content/gdrive

[5] DATA_PATH_FINAL = '/content/gdrive/MyDrive/MIDS/capstone'

[6] model = tf.keras.models.load_model(f"{DATA_PATH_FINAL}/tf_rand_val_nn_v1")
```

- **Output requirements**
 - Currently, the model output would be a floating number

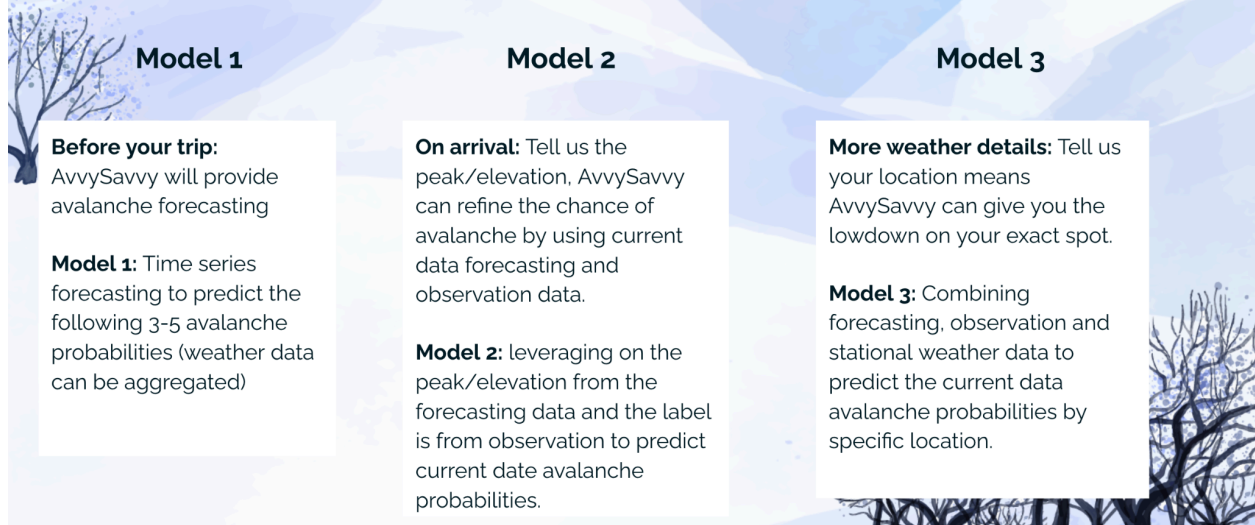
Feb 25, 2024

```
[ ] model.predict(ds)

/usr/local/lib/python3.10/dis
inputs = self._flatten_to_r
200/200 [=====]
array([[0.02322091],
       [0.02134966],
       [0.02007184],
       ...,
       [0.0121054 ],
       [0.01323311],
       [0.0124349 ]], dtype=f
```

1. Confirming our model deliverables (3 models) :
 - **Before your trip:** AvvySavvy will provide avalanche forecasting
 - **Field date:** Tell us the peak/elevation, AvvySavvy can predict avalanche probability on the field date.
 - **More weather details** mean AvvySavvy can give you the lowdown on your exact spot.

Model Approaches



2. 3 models:
 - a. Part 1: Time series forecasting: weather data can be aggregated (Sam)
 - i. <https://snowpack.slf.ch/>
 - b. Part 2: features in forecasting and observation (Luka)
 - i. 401 days in the 3 seasons
 - c. Part 3: features in forecasting and observation + station weather data (Luka)
 - i. $401 \times 3 = 1203$
 - ii.
3. UI Requirements (There will be 3 tabs, each will host one model?)
4. Part 1 Model
 - a. Date & date use
 - b. Lat & lng
 - c. Pulling the station weather live?
5. **Part 2 requirements in detail (Ravi): This tab requires users to input information(refer to input requirements), find additional features (refer to requirements), take the inputs and go through a model (refer to base model file), produce an output (output requirements).**
 - a. **Input Requirements:**
 - i. User inputs, the UI requires users to input: aspects, elevation, and date
 - ii. Aspects: create a drop-down for the available aspects:
 1. N
 2. NE
 3. E
 4. SE
 5. S
 6. SW

7. W
8. NW
- iii. Elevation: create a drop-down for the available elevation:
 1. Below Treeline
 2. Near Treeline
 3. Above Treeline
- iv. Date (Need discussion)
 1. Simple approach: we have to forecast data (features) until 1/30/2024, we saved our features in the table so that I can call the features from the same table.
 - a. Our model is only limited to 1/30/2024
 2. Complicated: if it's over 1/30/2024, we need to either scrap and save to the database or get an API to grab the feature?
 3. I think approach 1 makes more sense than our web app making the available data based on this table ([combined_date.csv](#)).
- b. **Additional features: since the model was trained more than the 3 features above, we need to save the additional features in the database, so that when the model is called, we can get the additional features:**
 - i. The additional features table can be referred to ([combined_features.csv](#))
- c. **Base Model File:**
 - i. [Model file: model_p2_v0.txt](#)
- d. **Output requirements**
 - i. Currently, the model output would be a floating number, we will need a logic in our API something like converting probabilities to binary predictions (0 and 1)
 1. $y_pred_binary = [1 \text{ if } pred \geq 0.07 \text{ else } 0 \text{ for } pred \text{ in } y_pred]$

Feb 21, 2024

Week 7

- Reminder: we need to do elevator speech for next week
- Peer review
- Deadline:
 - Data collection completion
 - Data combination with feature engineering
 - So that we can work on the model next week
- In the planning, we should start with the model development/experiment
- [Luka]
 - The observation + forecasting data is ready.
 - **Waiting to get** the weather data combined, so that I can do the EDA for the combined data, such as the correlation between snow depth and avalanche danger, wind, and problems like wind slab - The EDA with combined data would be the first portion of our presentation 2.

- Planning: I will do the feature engineering, most of our features are ready. I am experimenting with using LLM to extract more features from the observation summary.
- [Ravi] wanted to talk about the a sample demo web-app
- [Sam]
- Model development divisions among the team members:
 - Predict the likelihood of avalanches: Utilize weather data to estimate the probability of avalanches based on aspects, elevation, and size.
 - Forecasting the likelihood of avalanches by using weather time-series data: In contrast to Approach 1, which predicts today's avalanche likelihood, this method utilizes time-series information to forecast probabilities for tomorrow or the subsequent five days.
 - Predict the probabilities of avalanches within defined regions for more precise locations: Anticipate avalanche probabilities within specified regions: by deploying our model on a website where users input aspects, elevations, and which mountain, the model will provide the probabilities of avalanches.
 - Need specific weather data
 - Specific observational data

Feb 10, 2024

To do:

- ☒ ~~Luka: Cleaning last 3 seasons observation~~
- ☒ ~~Luka: Combining last 3 seasons' observations + last 3 seasons forecasting~~
- ☐ Ravi: Continue to get the current observations data
- ☐ Ravi: Combining current season's observation and current season forecasting
- ☐ Sam: Combining our avalanche data to weather data by date, radius of lat & lng

Our data g-drive:

<https://drive.google.com/drive/u/2/folders/1WBNsdlITJjvnCFTPlk4eCj0qANTwCO27>

Week 6 :

- **Objectives: data collection should be completed**
- **EDA completion**

Week 7: need to talk about ML algorithm

Tasks:

Data range:

- **Current season (Nov 29, 2023 - Jan 30, 2024)**
- **2022-23 season (Nov 26, 2022 - April 30, 2023)**
- **2021-22 season (Dec 1, 2021 - April 23, 2022)**

- 2020-21 season (Dec 12, 2020- April 18, 2021)

Training, validation -> last 3 seasons

Testing: current seasons

1. Forecasting data: completed
2. Observation data?
3. Weather data?

Next step:

1. Data combination

- a. [Ravi, Luka] Forecasting data & observation data: key -> date
 - i. Combining the last 3 seasons' observations data to current season data. For example, make sure the column name conversion is combined so that we can merge. Resolving conflict: For example, on Jan 1st, 2023, 5 observations on that date, 3 said avalanche observed while 2 said avalanche is not observed. prioritize: professional > forecaster > observer > public
- b. [Sam] Observation data & weather data: key -> date & lat & lng? (The current season doesn't have the lat & lng)


2. EDA on combination data - Next Week (Before our next presentation)

- a. Winds to windslab correlations
- b. Snow dump to avalanche danger scale

3. Data description extraction (Luka)

- a. Extract the information from the observation description

Feb 5, 2024

- Presentation  Avalanche Safety
- Data readiness?
 - Sam:
 - Historical weather
 - Trying on the forecasting data

•

Jan 31, 2024

- ☒ ~~Meet on Sunday 5 pm eventing~~
- ☒ ~~Luka start with our presentation deck~~

Jan 27, 2024

Action Item: The team member will work on getting the data first and making it to the tabular datasets that can be used for the ML. We will start to script each of the sites for Jan 2024 to ensure our script works before extending to 2023 data.

- ☒ ~~Luka - get the dataset from [Forecast](#)~~
- ☒ ~~Luka - will finish up the~~
~~[Team Project Plan template - Mountain Navigation Team](#)~~
- ☐ Ravi - Get the dataset from [Observation](#), for example, get everything from each of the observation tab:
<https://www.sierraavalanchecenter.org/observations#/view/observations/fbfbcb17a-753d-4936-b127-999aead0f551>
- ☐ Sam - get the dataset from the weather.gov or NOAA on hours history weather (Jan 2024) and forecasts by zipcode with station granularity

Time expectation: Data is crucial for the success of the project. We will have it due before our presentation week on Feb 7th.

Jan 25, 2024

Potential workflow:

1. During the navigation the day before: using approach 2 for forecasting
 2. On the field, take pictures of the mountains, and call Gemini to get general observation feedback
 3. Provide aspect and elevation, we can provide more details on the likelihood
 4. With more details weather info, we can provide more details on the specific location
- **[L] Three iterative approaches:**
 1. Predict the likelihood of avalanches
 2. Forecasting the likelihood of avalanches by using weather time-series data
 3. Predict the probabilities of avalanches within defined regions for more precise locations

Approach 1: Predict the likelihood of avalanches: Utilize weather data to estimate the probability of avalanches based on aspects, elevation, and size.

- **Training input data:**
 - Weather data from [Forecast](#)
- **Training label:**
 - Avalanche danger scale
- **Data:**
 - [Forecast](#)
- **Pros:**
 - Data is available
 - Easy to implement
- **Cons**

- Users need to input the weather data

Approach 2: Forecasting the likelihood of avalanches by using weather time-series data:
In contrast to Approach 1, which predicts today's avalanche likelihood, this method utilizes time-series information to forecast probabilities for tomorrow or the subsequent five days.

- **Training input data:**
 - Weather data from [Forecast](#), historical weather data
- **Training label:**
 - Avalanche danger scale
- **Data:**
 - [Forecast](#)
- **Pros:**
 - Data is available
 - Time-series
- **Cons**
 - Use need to input the past historical weather data

Approach 3: Predict the probabilities of avalanches within defined regions for more precise locations: Anticipate avalanche probabilities within specified regions: by deploying our model on a website where users input aspects, elevations, and which mountain, the model will provide the probabilities of avalanches.

- **Training input data:**
 - BACKCOUNTRY AVALANCHE FORECAST
 - Aspect, Elevation
 - Weather for a specific location
- **Training label:**
 - Observation (start from Sierra Avalanche Center), the label would be whether an avalanche is happening.
- **Data:**
 - [Forecast](#)
 - [Observations](#)
 - **Challenging points:** currently the forecast data only has general/broader weather, we need to get into NOAA to try to find weather data for a specific location
- **Pros:**

- Probabilities can be numeric, not the scale
- Straightforward inputs
- Features are simple
- Potentially two models
 - model 1 for predictions,
 - and model 2 to try **the LLM model to generate the training label.**
 - To extract aspect, elevation and avalanche probabilities from the observation, we can leverage PALM2 or GEMINI.
 - A PALM2 example:

```

1 prompt_1 = f"""
2 What is the instabilities or avalanche danger by the aspect and elevation given below.
3 Text: "{statement_1}"
4 Provide them in JSON format with following keys:
5 has_avalanche_danger, no_avalanche_danger.
6 Values should include aspect, elevation, and comments
7 """
8 response_1 = get_response(prompt_1)
9 print(response_1)

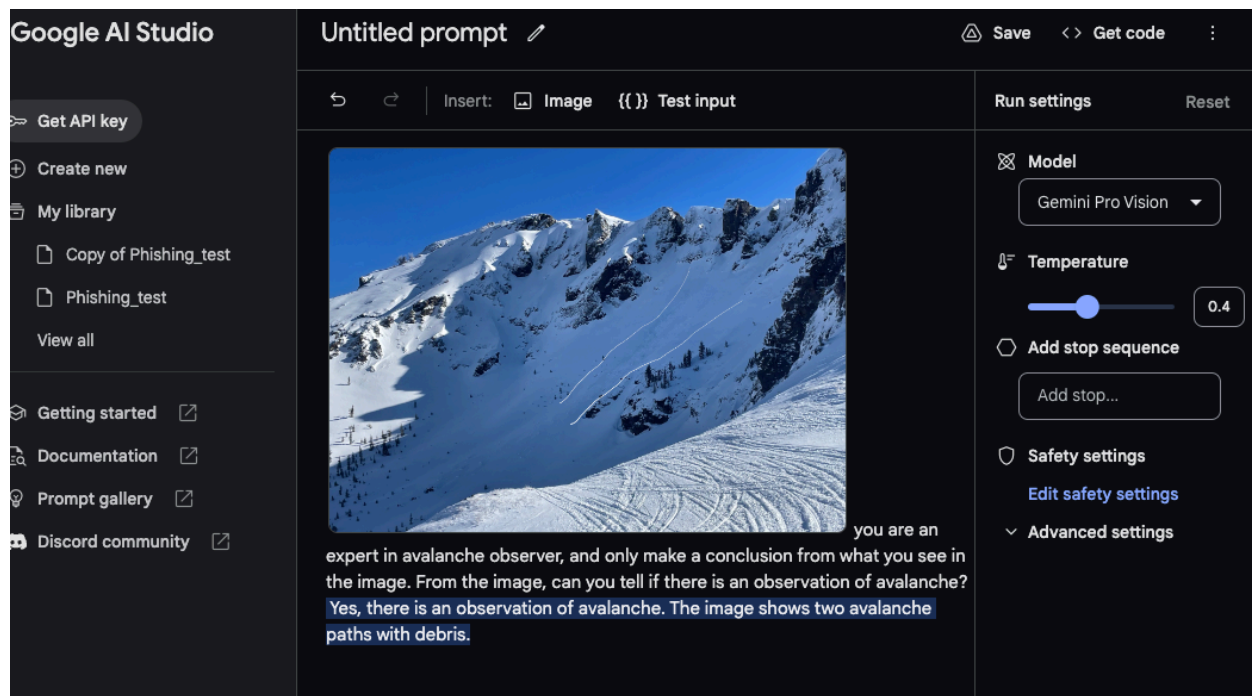
```

```

```JSON
{
 "has_avalanche_danger": [
 {
 "aspect": "South",
 "elevation": "9000'",
 "comments": "Snow bombs from melting ice and snow on trees"
 },
 {
 "aspect": "North",
 "elevation": "N/A",
 "comments": "No signs of instabilities from snowmobiler, skier, and snowboarder tracks"
 }
],
 "no_avalanche_danger": [
 {
 "aspect": "North",
 "elevation": "N/A",
 "comments": "No signs of instabilities from snowmobiler, skier, and snowboarder tracks"
 }
]
}
```

```

- With the Gemini, we can also load the images from the observations:



- **Cons**

- This functionality is restricted to the predefined regions, aligning with the areas used during model training.
- [Observations](#) are not standard, but also a potential opportunity for LLM.
- There might be conflicting data in the Observations dataset. For example, person 1 observed an avalanche aspect: N with elevation: above treeline can be contradicting against person 2, who observed the same aspect with the same elevation. Thus, we need to come up with a standard calculation for these conflicting opinions.
 - Can start with the Sierra Center
 - Then expand to the public