

Learn Models

Friday, 15 November 2024 2:40 PM

<https://developers.google.com/machine-learning/crash-course>

Linear regression :

Its based on formula $y = b + wx$

Where

b is the y intercept point

W is the slope / weight

X is the input data

If multiple things are into consideration then it will be

$Y = b + w_1x_1 + w_2x_2 \dots$

What parts of the linear regression equation are updated during training?

Its weight and bias

Recommendation System :

- Using cosine similarity
 - Formula = $A \cdot B / |A| |B|$
 - E.g if $A = \{1,2,0,4\}$, $B = \{2,2,2,1\}$
 - $A \cdot B = 1*2 + 2*2 + 0*2 + 4 * 1 = 10$
 - $|A| = \sqrt{1^2 + 2^2 + 0^2 + 4^2} = 4.58257569495584$
 - $|B| = \sqrt{2^2 + 2^2 + 2^2 + 1^2} = 3.60555127546399$
 - $A \cdot B / |A| |B| = 10 / (4.582 * 3.605) = 0.6054$
 - Which is 60.54 % similar

There are several machine learning models you can use for the Titanic survival prediction problem. Each has strengths and weaknesses depending on the data and problem complexity. Here are some common models:

1. Logistic Regression

- Suitable for binary classification problems.
- Simple and interpretable.
- Example:

```
python
Copy code
from sklearn.linear_model import LogisticRegression
model = LogisticRegression()
model.fit(X, y)
predictions = model.predict(X_test)
```

2. Decision Tree

- Splits data into smaller subsets based on feature values.
- Easy to visualize.
- Example:

```
python
Copy code
from sklearn.tree import DecisionTreeClassifier
model = DecisionTreeClassifier(max_depth=5, random_state=1)
model.fit(X, y)
```

3. Random Forest

- Ensemble of decision trees (used in your example).
- Robust to overfitting.
- Example:

python

```
python
Copy code
from sklearn.ensemble import RandomForestClassifier
model = RandomForestClassifier(n_estimators=100, max_depth=5, random_state=1)
```

4. Gradient Boosting (e.g., XGBoost, LightGBM, CatBoost)

- Builds trees sequentially to minimize errors.
- Performs well on structured/tabular data.
- Example (XGBoost):

```
python
Copy code
from xgboost import XGBClassifier
model = XGBClassifier(max_depth=5, n_estimators=100, random_state=1)
model.fit(X, y)
```

5. Support Vector Machine (SVM)

- Finds the optimal hyperplane to separate data classes.
- Example:

```
python
Copy code
from sklearn.svm import SVC
model = SVC(kernel='linear')
model.fit(X, y)
```

6. Neural Networks

- Suitable for complex patterns or non-linear relationships.
- Example (using Keras):

```
python
Copy code
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense

model = Sequential([
    Dense(64, activation='relu', input_shape=(X.shape[1],)),
    Dense(1, activation='sigmoid')
])
model.compile(optimizer='adam', loss='binary_crossentropy', metrics=['accuracy'])
model.fit(X, y, epochs=10, batch_size=32)
predictions = (model.predict(X_test) > 0.5).astype(int)
```